MASTER'S DEGREE THESIS

MSc in Mathematics Engineering

Title: Obtaining gravitational models from planetary ephemeris

Author: Patricia Sánchez Martín

Advisor: Josep J. Masdemont

Department: Matemática Aplicada I

Academic year: 2011-2012

Facultat de Matemàtiques i Estadística

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Universitat Politècnica de Catalunya
Facultat de Matemàtiques i Estadística

Master's Degree Thesis

Obtaining gravitational models from planetary ephemerides

Patricia Sánchez Martín

Advisor: Josep J. Masdemont

Matemática Aplicada I

Abstract

Keywords: Ephemerides, software ephemerides JPL, coordinate systems

Ephemerides describe the position and velocity of a celestial object as a function of time. The ephemerides data are obtained by solving the fundamental equations of motion of the body, which is obtained by applying the fundamental laws of motion postulated by Keppler and Newton. The objective of this project is to build a software package that manages JPL ephemerides. With this software, we will can select planets or natural satellites contained in the ephemerides file and we will can work with them on different ways. Also, we are going to do several programs to make changes of coordinates and some vector fields. Finally, we are going to explain an application of this software, such as the parallel shooting, and we are going to show different simulations.

Contents

Chapter 1
Introduction

1. Coordinate Systems

In astronomy, it is necessary and convenient to represent the position of an object, such as a star or a planet, in several different coordinate systems, according to the context in which the position is to be used. Each coordinate system corresponds to a particular way of expressing the position of a point with respect to a coordinate frame. Often, coordinate frames with respect to which observations are made, differ from the coordinate frame that is most convenient for the comparison of observational data with theory. In general, each position is represented by a set of three coordinates that specify the position of the object with respect to a particular coordinate frame. In many cases the distance of the object is inknown; so two coordinates are sufficient to represent the direction to the object, although three direction cosines may be used.

In general, an object is moving with respect to the coordinate frame, and the coordinate frame is moving and rotating in "space". Moreover, positions may be of several kinds, according to whether or not allowances have been made for aberration, diurnal rotation, refraction and other factors that affect the direction in which an object is observed. Ephemerides (or ephemeris), that represent high-precision positions and properties of astronomical objects at given times, must always be accompanied by a precise statement that specifies all of these various factors.

To design the coordinates, different coordinate systems can be used to specify the positions of celestial objects. Each system depends on the choice of coordinate frame and on the way of specifying coordinates with respect to the frame. The designations used to indicate the principal origins of celestial coordinate frames are as follows:

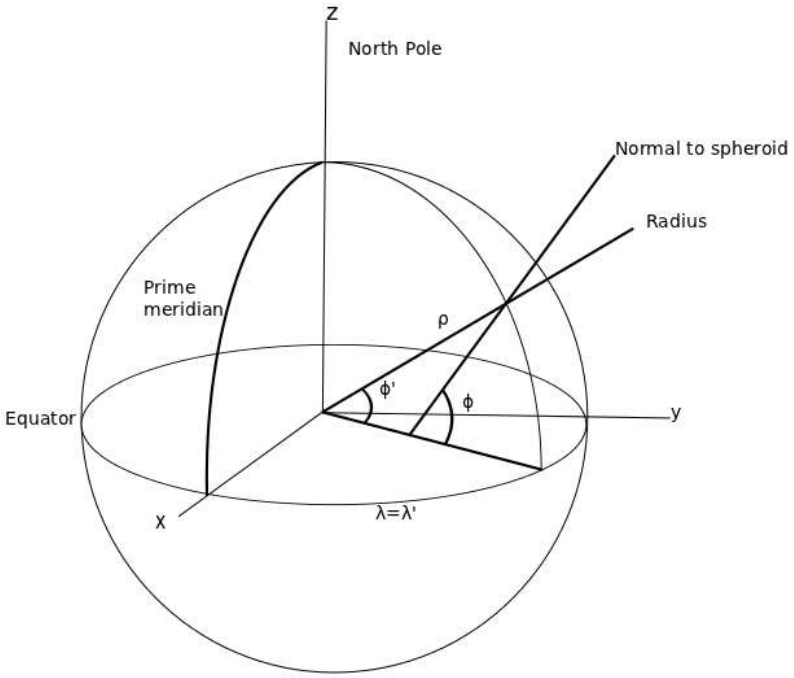

FIG. 1. Geocentric (λ', ϕ', ρ) and geodetic (λ, ϕ) coordinates.

(1) Topocentric: viewed or measured from the surface of the Earth.
(2) Geocentric: viewed or measured from the center of the Earth.
(3) Selenocentric: viewed or measured from the center of the Moon.
(4) Planetocentric: viewed or measured from the center of the planet (with corresponding designations for individual planets).
(5) Heliocentric: viewed or measured from the center of the Sun.
(6) Barycentric: viewed or measured from the center of mass of the solar system (or of the Sun and a specified subset of planets).

The principal celestial reference planes through the appropriate origins are as follows:

(1) Horizon: the plane that is normal to the local vertical (or apparent direction of gravity) and passes through the observer.
(2) Local meridian: the plane that contains the local vertical and the direction of the axis of rotation of the Earth.
(3) Celestial equator: the plane that is normal to the axis of the rotation of Earth and passes through the Earth's center.

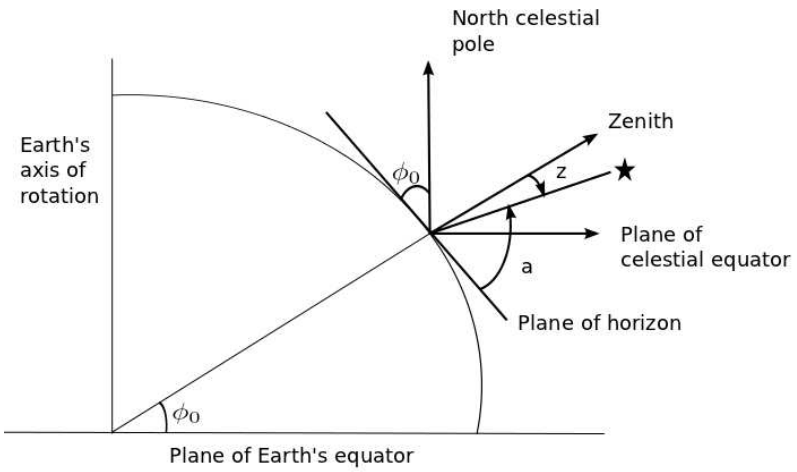

FIG. 2. Relation between geographic latitude and the altitude of the celestial pole.

(4) Ecliptic: the mean plane (i.e., ignoring periodic perturbations) of the orbit of the Earth around the Sun.
(5) Planet's meridian: a plane that contains the axis of rotation of the planet and passes through the observer.
(6) Planet's equator: the plane that is normal to the axis of rotation of the planet and passes through the planet's center.
(7) Orbital plane: the plane of the orbit of a body around another (e.g., of a planet around the Sun or barycenter).
(8) Invariable plane or *Laplacian plane*: the plane that is normal to the axis of angular momentum of a system and passes through its center.
(9) Galactic equator: the plane through the central line of the local Galaxy (Milky Way).

2. Timescales

One can differentiate between: Atomic Timescales, Dynamical Time, Rotational Timescales and Coordinated Universal Time.

First, we are going to talk about **Atomic Timescales**. For scientific, practical and legal purposes, the standard unit for the measurement of intervals of time is the System International (SI) second, which is defined by the adoption of a fixed value for the frequency of a particular transition of cesium atoms. Time can be measured in this unit by the use of time standards based on physic processes. Such a timescale provides a measure

of time for identifying the instants at which events occur, the interval of time between two events can be calculated as the differences between the times of the events. The results of the intercomparison of about 200 frequency standards located around the world are combined to create a standard timescale that can be used for identifying uniquely the instants of time at which events occur on the Earth. This standard timescale is known as International Atomic Time (TAI) and it is the basis for all timescales in general use. It is distributed by many different means, including radio time signals, navigation systems, communication satellites, and precise time standards.

It must also take in mind relativistic effects. In high-precision timekeeping, and for some purposes in solar-system dynamics, it is necessary to take in mind the effects of special and general relativity and to recognize, for example, the rate of an atomic clock depends on the gravitational potential in which it is placed and that the rate will appear to depend on its motion relative to another clock with which it is compared. In particular, one should recognize that the independent variable, or timescale, of the equations of motion of the bodies of the solar system depends upon the coordinate system to which the equations refer.

Now, we are going to talk about **Dynamical Time**. First, it is noteworthy that equations of motion of the bodies of the solar system involve time as an independent variable. Until 1960, was used mean solar time, but when it was discovered that the rotation of the Earth was irregular, a new timescale was introduced that corresponded to the independent variable. This was called Ephemerides Time (ET) and was based on the motion of the Sun. Later, when it was necessary to distinguish between relativistic effects (which cause differences between timescales for the center of the Earth and the center of the solar system) the dynamical time arguments were introduced (Terrestrial Dynamical Time, TDT, and Barycentric Dynamical Time, TDB). TDT was defined in a way that keeps continuity with ET. Since ET was not specified as either TDT or TDB, either can be considered to be the extension of ET. Since TDT is defined in terms of TAI, which can be determined only back to 1956, the determination of dynamical time prior to 1956 must be based in comparison of observations and theories of the motions of the Sun, Moon and planets.

On the other hand, TDT is used as the timescale for the geocentric ephemerides (giving apparent positions with respect to the center of the Earth) and TDB is used as

the timescale for the ephemerides that refer to the center of the solar system. TDT differs from TAI by a constant offset, which was chosen arbitrarily to give continuity with ephemerides time. TDB and TDT differ by small periodic terms that depend on the form of the relativistic theory is being used.

Also, we are going to explain a little about **Rotational Timescales**. As the Earth rotates about its axis, it also moves in its orbit around the Sun. Thus, while the Earth rotates once with respect to a fixed star, the Earth moves in its orbit so that additional rotation is necessary with respect to the Sun. The rotation of the Earth with respect to the equinox is called sidereal time (ST). The Earth's rotation with respect to the Sun is the basis of Universal Time, also called solar time. Since the rotation of the Earth is attached to irregular forces, sidereal time and Universal Time are irregular with respect to atomic time. Sidereal time is the hour angle of the catalog equinox and is attached to the motion of the equinox itself due to precession and nutation. Otherwise, it is a direct measure of the diurnal rotation of the Earth. Sidereal time reflects the actual rotation of the Earth and can be determined by observations of stars, artificial satellites and extragalactic radio sources. On the other hand, the apparent diurnal motion of the Sun involves both the non-uniform diurnal rotation of the Earth and the motion of the Earth in the orbit around the Sun. In practice, Universal Time is directly related to sidereal time by means of a numerical formula. For each local meridian there is a corresponding *local sidereal time*. The measure of the rotation of the Earth with respect to the true equinox is called *apparent sidereal time*. The measure with respect to the mean equinox of date is referred to as *mean sidereal time*. Apparent sidereal time minus mean sidereal time is called the *equation of the equinoxes*.

Finally, we can explain the **Coordinated Universal Time** (UTC). Although TAI provides a continuous, uniform and precise timescale for scientific reference purposes, it is not convenient for general use. In everyday life it is more convenient to use a system of timescales that correspond to the alternation of day and night, and can be easily related to each other and to TAI. In this timescales, the numerical expression or measure, of the time of an event, is given in the conventional form of years, months, days, hours, minutes, seconds and decimals of seconds. The standard time on the prime meridian is known as UTC. UTC is an atomic timescale that is kept in close agreement with Universal Time

(UT), which is a measure of the rotation of the Earth on its axis. The rate of rotation of the Earth is not uniform (with respect to atomic time), and the difference between TAI and UT is increasing irregularly by about one second every eighteen months. The difference between UTC and TAI is always an integral number of seconds. UTC is maintained in close agreement with UT introducing extra seconds (which is better than one second). This extra seconds, know as *leap seconds*, are added to UTC, usually at the end of the last day of June or December.

3. The Enumeration of Dates

There are three ways to enumerate days that could be taken: Civil Calendar, Julian Date, and Besselian and Julian Years (these last two are the most interesting for astronomical purposes).

Civil calendar is the calendar that everyone knows, so we just say that was derived from the Julian calendar.

For many astronomical purposes is more convenient to use the **Julian Date** (JD), because this represents a continuous count of days that is known as the Julian day number, and the day number is extended by the addition of the time of day, expressed as a decimal fraction of day. It is important to note that the integral values of the Julian date refer to the instants of Greenwich mean noon; correspondingly, the Julian date for 0^h UT always ends in ".5". This, and the fact that current Julian dates require seven digits to express the integral parts, make the Julian date system unsuitable for some purposes. The value of JD minus 2400000.5 is sometimes used for current dates; it is known as the modified Julian date (MJD) (but as there are some more kinds of MJD, different values are substracted to calculate different MJD). The Julian date system may be used with TAI, TDT or UTC, so when the difference is significant, the particular timescale should be indicated.

If we want to specify the epoch of celestial coordinate system, it is more appropriate to use **Besselian and Julian Years**, because it is convenient to measure time in years and to identify an instant of time by giving the year and the decimal fraction of the year to a few places. This system was introduced by Bessel, and is still in use, but

it has two disadvantages. First, the length of the year varies slowly, and second, the instants at the beginning of the years (.0) do not correspond to Julian dates, which are convenient for use in dynamical astronomy. The Besselian system has been replaced by a new system in which 100 years is exactly 36525 days (or 1 Julian century) and in which 1900.0 corresponds exactly to the epoch 1900 January 0.5, from which time interval was reckoned in the principal theories of the motions of the Sun, Moon and planets. The old and new systems are distinguished when it is necessary by the use of the prefix letters "B" and "J".

4. Ephemerides

Ephemerides describe the position and velocity of a celestial object as a function of time. The ephemerides data are obtained by solving the fundamental equation of motion of the body, which is obtained by applying the fundamental laws of motion postulated by Keppler and Newton. There are two basic ways to solve the fundamental equation of motion to obtain ephemerides: an analytical and a numerical way.

- Analytical ephemerides are based on closed-form algebraic expressions that give the object's position and velocity components for a given instance of time. These expressions must be derived from an algebraic solution to the equation of motion for the object. The primary benefit of analytical ephemerides is that they express the position and velocity components as explicit functions of time. When mutual gravitational perturbations and relativistic effects are taken in mind, the expressions necessarily become quite complicated and analytical solutions may become unfeasible. In general, for applications requiring a certain degree of precision, analytical ephemerides are no longer used.
- Numerical ephemerides are based on a numerical solution of the equation of motion. The output of such a computation is a table of numbers giving the position and velocities at a desired instances of time. A potential drawback of this method is the sheer size of the tables when the object's position and velocity components are required for a large number of times. In practice, the numerically generated position tables (and velocities) may be compressed by ”fitting” them with a mathematical function, which can replicate the original table values to within a very small tolerance for any desired instance of time. This is the approach taken e.g. in the production of the Jet Propulsion Laboratory (JPL) planetary ephemerides. Notice that the ”fitting”

function, which allows - a part of the - ephemerides to be calculated in an algebraic way is not the same as the algebraic solution to the fundamental equation of motion.

We will use these last ephemerides, concretely JPL ephemerides. JPL ephemerides are generally created to support spacecraft missions to the planets. Selected ephemerides are recommended for more general use, in particular DE405 which is the basis for the Astronomical Almanac. Other ephemerides generated for specific mission requirements may be available for use but usually without documentation available to outside users.

The JPL ephemerides are saved as files of Chebyshev polynomials fit to the Cartesian positions and velocities of the planets, Sun, and Moon, typically in 32-day intervals. The positions are integrated in astronomical units (AU), but with polynomials stored in units of kilometers. The integration time units are days of barycentric coordinate time (TDB). The value of the astronomical unit is estimated based from measurements of planetary orbits using the Gaussian gravitational constant k as adopted by the International Astronomical Union. The mass parameter (GM) of the Sun is related to the estimated AU by the relation GM_{sun} = k·k/AU, where k is the Gaussian gravitational constant. Thus a change in the value of the AU corresponds to a change in the GM of the Sun in SI units. Since the ephemerides fits are sensitive to the GM of the Sun, each ephemerides generally has a different estimate of the AU. The mass parameters (GM) of the planets are best determined by tracking of spacecraft in orbit about or encountering the planets. Each ephemerides uses the best estimates of the planetary GM values available at the time.

Most JPL ephemerides files include Chebyshev polynomials fit to the lunar libration angles, which are integrated along with the planetary positions. Many ephemerides files also have a fit to the 1980 IAU nutation series. While the 1980 IAU nutation series is not current, it is maintained in the files for backward compatibility.

To access the JPL ephemerides there are several ways. For users who need positions of planets at a few specific time, JPL's interactive website and telnet service, "Horizons", provides a wide variety of astronomical information including planetary positions from the planetary ephemerides DE405, at web site

http://ssd.jpl.nasa.gov/?horizons.

For users who need the capability of looking up the positions of planets or satellites at many times or as needed for other computations, the recommended means of reading the ephemerides is through use of the SPICE software toolkit,

http://naif.jpl.nasa.gov/naif/index.html.

The SPICE software is supported in several programming languages (including C, Fortran and Matlab) on multiple platforms and compilers. The SPICE software reads JPL ephemerides in a machine-independent binary format (kernels) which are available from the SPICE web site and by anonymous FTP from

ftp://naif.jpl.nasa.gov/pub/naif/generic_kernels/spk/.

The JPL planetary ephemerides are also available in an ASCII format which can be converted by the user to machine-dependent binary files for reading. The ASCII files are created in blocks of 20 or more years. Later ephemerides tend to be stored in longer blocks. When ephemerides in ASCII format are converted, the files can be merged to span a longer time if desired. The ASCII files are available by anonymous ftp from

ftp://ssd.jpl.nasa.gov/pub/eph/planets/ascii/.

The ASCII file format is briefly described in the file

ftp://ssd.jpl.nasa.gov/pub/eph/planets/ascii/ascii_format.txt.

The machine-dependent binary files differ mainly in the binary representation of numbers. The 'little-endian' representation is used by Intel x86, IA64 or Alpha based processors (and their clones). Some binary ephemerides files are available in this format from

ftp://ssd.jpl.nasa.gov/pub/eph/planets/Linux.

The 'big endian' representation is used by PowerPC and SPARC processors. Some binary files in this format are available from

ftp://ssd.jpl.nasa.gov/pub/eph/planets/SunOS.

Fortran programs for converting the ASCII files and for merging and reading the binary files are available by anonymous ftp from

ftp://ssd.jpl.nasa.gov/pub/eph/planets/fortran.

These Fortran programs generally require some adaptation by the user. Instructions for using these programs are available for downloading from

ftp://ssd.jpl.nasa.gov/pub/eph/planets/fortran/userguide.txt.

Other programs for reading these formats have been developed independently by outside programmers, and not suported directly by JPL, are listed at

ftp://ssd.jpl.nasa.gov/pub/eph/planets/other_readers.txt.

In this project, we are going to use the JPL ephemerides DE406. These ephemerides are constructed with the same integration as DE405, with the accuracy of the interpolating polynomials has been lessened to reduce file size for the longer time span covered by the file. These ephemerides include neither nutations nor librations. Coordinates of the Sun, Moon and the nine major planets can be calculated for dates between JED (Julian Ephemerides Date) 0624976.50 (-3001 Feb 4) and JED 2816912.50 (+3000 May 5) referring to the International Celestial Reference Frame (ICRF).

Also, we have used different `*.bsp` files regarding to some natural satellites of the planets. These files contains information about natural satellites of the planets. As the ephemerides DE405, DE406 and `*.bsp` files for the natural satellites are similar, and contains similar information, we will explain the format of the DE405 because there is more documentation about these ones.

5. Ephemerides DE405

We are going to talk about the fundamental planetary and lunar ephemerides DE405 from Jet Propulsion Laboratory (JPL). DE405 result from a least-squares adjustment of a previously existing ephemerides to a variety of observational data (measurements), followed by a numerical integration of the dynamical equations of motion which describe the gravitational physics of the solar system.

For the final phase of integration of the ephemerides creation process, it is necessary each one of this phases:

- The equations of motion describing the gravitational physics which govern the dynamical motions of the bodies.
- A method for integrating the equations of motion.
- The starting positions and velocities of the bodies at some initial epoch along with the values for various constants which affect the motion (e.g., planetary masses).

It is mainly the accuracy of the third component, the initial conditions and dynamical constants, which determines the accuracy of modern-day ephemerides, since the other two components (the physics and the integration method) are both believed to be sufficiently complete and accurate. The values of the initial conditions and constants are determined by the least squares adjustment of them to the observational data (measurements), and the accuracy of this adjustment, and thus of the ephemerides themselves, depends primarily upon the distribution, variety, and accuracy of the observational data.

It is assumed that the equations of motion accurately describe the basic physics which govern the motion of the major bodies of the solar system - at least to the presently observable accuracy. For the motion of the Moon, the fitting of the lunar ephemerides to the lunar laser-ranging observations is used to estimate the constants involved and to help distinguish various models of the lunar interior, Earth-raised tides, etc.

Numerical integration of the equations of motion is the only known method capable of computing fundamental ephemerides at an accuracy which is comparable to that of the modern-day observations; analytical theories have not been able to attain such high accuracy. The computer program which was used to integrate the equations of motion for DE405 has been demonstrated to be suficiently accurate ([**8**]). And the reference frame for the ephemerides is the International Celestial Reference Frame (ICRF [**7**]).

The independent variable of the equations of motion, and, thus, of the fundamental ephemerides themselves, may be termed, "T_{eph}". It is a fully rigorous relativistic coordinate time, implicitly defined by the equations of motion themselves. T_{eph} may be considered similar to previously-defined "ET" or to "TDB", since their average rates are the same, even though there are basic differences in the definitions. The time recently defined by the IAU, "TCB", is mathematically equivalent to T_{eph}; TCB and T_{eph} differ by only a constant rate.

6. Constructing ephemerides DE405

To construct the ephemerides file first we will explain equations of motion. These equations describe the forces upon the planets, Sun and Moon which affect their motions and the torques upon the Moon which affect its orientation. It believes that given the accuracy of the observations, there is nothing to suggest that other forces

or different forces are present in the Solar System. The uncertainties existing in the planet's and Moon's motions are certainly explainable, considering the uncertainties in the observations and in the fitted initial conditions and dynamical constants.

The equations of motion used for the creation of DE405 included contributions from: (a) point-mass interactions among the Moon, planets, and Sun; (b) general relativity (isotropic, parametrized post-Newtonian); (c) Newtonian perturbations of selected asteroids; (d) action upon the figure of the Earth from the Moon and Sun; (e) action upon the figure of the Moon from the Earth and Sun; (f) physical libration of the Moon, modeled as a solid body with tidal and rotational distortion, including both elastic and dissipational effects; (g) the effect upon the Moon's motion caused by tides raised upon the Earth by the Moon and Sun, and (h) the perturbations of 300 asteroids upon the motions of Mars, the Earth, and the Moon.

For the point-mass interactions one can say that, the principal gravitational force on the nine planets, the Sun, and the Moon is modeled by considering these bodies to be point masses in the isotropic, parametrized post-Newtonian (PPN) n-body metric ([**11**]). The n-body equations were derived by Estabrook ([**2**]) from the variation of a time-independent Lagrangian action integral formulated in a non-rotating solar system barycentric Cartesian coordinate frame. Also are included Newtonian gravitational perturbations from 300 asteroids, chosen because they have the most pronounced effect on the Earth-Mars range over the time span covered by the accurate spacecraft ranging observations.

For each body `i`, the point-mass acceleration is given by

(1)

$$\begin{aligned}\ddot{r}_{i_{pointmass}} = &\sum_{j\neq i}\frac{\mu_j(r_j-r_i)}{r_{ij}^3}\{1-\frac{2(\beta+\gamma)}{c^2}\sum_{k\neq i}\frac{\mu_k}{r_{ik}}-\frac{(2\beta-1)}{c^2}\sum_{k\neq j}\frac{\mu_k}{r_{jk}}+\gamma\left(\frac{v_i}{c}\right)^2\\ &+(1+\gamma)\left(\frac{v_j}{c}\right)^2-\frac{2(1+\gamma)}{c^2}\dot{r_i}\cdot\dot{r_j}-\frac{3}{2c^2}\left[\frac{(r_i-r_j)\cdot\dot{r_j}}{r_ij}\right]^2+\frac{1}{2c^2}(r_j-r_i)\cdot\ddot{r_j}\}\\ &+\frac{1}{c^2}\sum_{j\neq i}\frac{\mu_j}{r_{ij}^3}\{[r_i-r_j]\cdot[(2+2\gamma)\ddot{r}_i-(1+2\gamma)\dot{r_j}]\}(\dot{r}_i-\dot{r}_j)\\ &+\frac{3+4\gamma}{2c^2}\sum_{j\neq i}\frac{\mu_j\ddot{r}_j}{r_{ij}}+\sum_{m=1}^{3}\frac{\mu_m(r_m-r_i)}{r_{im}^3}+\sum_{c,s,m}F\end{aligned}$$

where $\mathtt{r}_i$,$\dot{r}_i$, and $\ddot{r}_i$ are the solar system barycentric position, velocity and acceleration vectors of body $\mathtt{i}$; $\mu_j = Gm_j$, where $\mathtt{G}$ is the gravitational constant and $\mathtt{m}_j$ is the mass of body $\mathtt{j}$; $r_{ij} = |r_j - r_i|$; β is the PPN parameter measuring the nonlinearity in superposition of gravity; γ is the PPN parameter measuring space curvature produced by unit rest mass (in this integration, as in general relativity, $\beta = \gamma = 1$); $v_i = |\dot{r}_i|$; and $\mathtt{c}$ is the velocity of light.

The quantity $\ddot{r}_j$ appearing in two terms on the right-hand side of 1 denotes the barycentric acceleration of each body $\mathtt{j}$ due to Newtonian effects of the remaining bodies and the asteroids. Thus, the right hand side of the equation is dependent upon the left hand side, and so, rigorously, the computation should be iterated. However, use of Newtonian accelerations for these terms is sufficiently accurate.

In the next to last term on the right-hand side of 1, quantities employing the index $\mathtt{m}$ refer to the "Big3" asteroids, Ceres, Pallas, and Vesta. The last term represents forces upon the Earth, Moon, and Mars, only, from 297 other asteroids, grouped according to three taxonomic classes (C, S, M).

If we talk about solar system barycenter, it must take in mind that, in the $\mathtt{n}$-body metric, all dynamical quantities are expressed with respect to a center of mass whose definition is modified from the usual Newtonian formulation. The solar-system barycenter is given by Estabrook ([**3**]) as

$$\sum_i \mu_i^* r_i = 0 \tag{2}$$

where

$$\mu_i^* = \mu_i \left\{ 1 + \frac{1}{2c^2} v_i^2 - \frac{1}{2c^2} \sum_{j \neq i} \frac{\mu_j}{r_{ij}} \right\} \tag{3}$$

and where μ_i is the gravitational constant times the mass of the ith body; v_i is the barycentric velocity of the $\mathtt{i}^{th}$ body, and

$$r_{ij} = |r_j - r_i| \,. \tag{4}$$

During the process of numerical integration, the equations of motion for only the Moon and planets were actually evaluated and integrated. The barycentric position and velocity of the Sun were obtained from the equations of the barycenter. It should be noted that each of the two barycenter equations depends upon the other, requiring an iteration

during the evaluation of the solar position and velocity.

For the figure effects, long-term accuracy of the integrated lunar orbit requires the inclusion of the figures of the Earth, Moon, and Sun in the mathematical model. In DE405 the gravitational effects due to figures include:

1) The force of attraction between the zonal harmonics (through fourth degree) of the Earth and the point-mass Moon, Sun, Venus and Jupiter.
2) The force of attraction between the zonal harmonics (through fourth degree) and the second-through fourth-degree tesseral harmonics of the Moon and the point-mass Earth, Sun, Venus, and Jupiter.
3) The dynamical form-factor of the Sun ($\mathtt{J}_2$).

The contribution to the inertial acceleration of an extended body arising from the interaction of its own figure with an external point mass is expressed in the $\xi\eta\zeta$ coordinate system, where the ξ-axis is directed outward from the extended body to the point mass; the $\xi\zeta$-plane contains the figure (rotational) pole of the extended body, and the η-axis completes the right handed system.

7. The Numerical Integration

The calculation and arrangement of accelerations at each integration step is as follows:

1) The integrator subroutine provides new states (positions and velocities) for the nine planets and the Moon.
2) The asteroid states are evaluated from fixed polynomials.
3) Equations for the relativistic masses are evaluated for the planets, Moon, asteroids, and Sun, using current states for all bodies except the Sun. (The barycentric state of the Sun calculated at the end of the previous step is retained for this evaluation.)
4) The present approximate state of the Sun is obtained from the constraint (Equation 2).
5) Equations 3 are evaluated again, using this new estimate of the solar state.
6) Equation 2 is evaluated a second time to provide the current state of the Sun.
7) Equation 1 is evaluated to obtain the accelerations of the nine planets and the Moon.

8) It has been proved numerically more suitable to integrate the lunar ephemerides relative to the Earth rather than to the solar system barycenter. The solar system barycentric Earth and Moon states are replaced by the quantities $\mathtt{r}_{em}$ and $\mathtt{r}_B$ given by

$$r_{em} = r_m - r_e \tag{5}$$

and

$$r_B = \frac{\mu_e r_e + \mu_m r_m}{\mu_e + \mu_m} \tag{6}$$

where the subscripts $\mathtt{e}$ and $\mathtt{m}$ denote the Earth and Moon, respectively. Note that $\mathtt{r}_{em}$ is the difference of solar system barycentric vectors and is distinguished from a geocentric vector by the relativistic transformation from the barycenter to geocenter. (The vector $\mathtt{r}_B$ can be interpreted as a representation the coordinates of the Newtonian Earth-Moon barycenter relative to the solar system barycenter. It has no physical meaning and does not appear in force calculations; it is only a way for improving the numerical behavior of the differential equations.)

Respect to the estimated integration error, the method of error control used in the integration puts a limit on the absolute value of the estimated error in velocity of each equation at the end of every integration step. Step size and integration orders were adjusted on the basis of estimated error. The limits selected for DE405 were $2 \cdot 10^{-17}$ *au/day* in each component of the equations of motion for the planets and Moon, and $2 \cdot 10^{-15}$ *rad* d^{-1} for each component of the libration equations.

Integrations before to DE405 were performed on a Univac mainframe computer in double precision, with a 60-bit mantissa; DE405 was integrated on a VAX Alpha in quadruple precision. In all cases, the integration error has been significantly less than the estimated error resulting from the uncertainties in the adjustment of the initial conditions and constants to the observational data.

8. DE406, the new "JPL long ephemerides"

The full precision numerical integration covered the interval 3000 BC to 3000 AD. However, only the interval, 1600 AD to 2200 AD, has been fit with full precision Chebychev polynomials; this set of polynomials is referred to as DE405. DE406, on the other hand, covers the full interval of the integration, but in order to conserve disk

space, lower order Chebychev polynomials are used and both the nutations and the lunar librations have been excluded from the file. For DE406, however, the interpolation on the 64 day mesh points remains exact, and for other times, the interpolating accuracy is no worse than 25 meters for all planets and no worse than 1 meter for the Moon (adequate for nearly any application except for the processing of the most accurate observations). The nutations may be computed from the formula of the 1980 IAU Nutation Theory, and the librations have been saved on a separate file.

The binary version of DE406 occupies only 3.3 megabytes per century as opposed to the 9.2 required by DE405. For the full expansion, 3000 BC to 3000 AD, DE406 occupies only 200 megabytes.

Chapter 2
SPICE library

To read the file of ephemerides DE406 and the files of satellites, we need to use the library, in JPL web, ”SPICE” because this files are storage in a special binary (extension *.bsp).

SPICE stores data in files that are often referred to as ”kernels”. A kernel may store data in either text (ASCII) or binary format. In order to access data within a kernel an application program must ”load” the kernel using a SPICE API (”furnsh”).

The set of binary kernels consists of:

- SP-kernels (SPK). (We are using this one).
- Binary-style (just a few) PC-kernels (PCK).
- C-kernels (CK).
- Some spacecraft Events kernels (EK/ESP).

Binary kernels typically contain numeric and textual information. Hardware architecture (the CPU chip) determines the format of numeric binary data; the two formats used by computers supported by NAIF are called ”big endian” and ”little endian”. Because kernels are frequently transferred between computers which may use incompatible binary architecture, SPICE software has been designed to read non-native binary kernels. (But not to write to a non-native binary kernel).

There are three classes of kernels, but we are only using one, ”generic kernels”. Generic kernels are those that are not tied specifically to one mission; they are usually applicable to many missions, or could be useful independent of any mission. Examples of generic kernels are:

- The planetary, satellite, comet and asteroid ephemerides files produced by JPL (SPKs).
- The leapseconds kernel (LSK).
- The ”generic” version of the planetary constants kernel (text PCK).
- The high-precision lunar orientation kernel (binary PCK).
- Earth station topocentric locations (SPK) and reference frames (FK).
- Various kinds of mission-independent reference frame specifications (FK).

1. SPICE ephemerides subsystem: SPK files

The SPK system is the component of SPICE concerned with ephemerides data. Since SPK files are binary files, we can't just open them with a text editor to determine which ephemerides objects are represented in the file. Instead we need to use one of the SPICE utility programs that allow to summarize the ephemerides contents of an SPK file. Also, SPICE kernels may contain metadata that describes the contents, intended use, accuracy, etc. of the kernel. This metadata is called the ”comments” portion of the kernel. Many SPK files contain comments that can help to decide upon the suitability of the kernel for the application.

An SPK file contains ephemerides (trajectory) data for ”ephemerides objects” (position and velocity of objects as a function of time). Spacecraft, planets, satellites, comets and asteroids are the obvious kinds of ”ephemerides objects”, but many other possibilities exist, such as: a rover on the surface of a body, a camera on top of a mast on a lander, a transmitter cone on a spacecraft, a deep space communications antenna on the Earth, the center of mass of a planet/satellite system (planet barycenter), the center of mass of our solar system (solar system barycenter). To work with SPK files, we need to take in mind the following terminology:

- Reference Frame: A reference frame is a Cartesian coordinate system with three axes: `x`, `y` and `z`. The axes are mutually orthogonal. The center of the frame is the origin of the Cartesian reference system. For the reference frames in SPICE, the positions of the axes are typically defined by some observable object. For example, in the `J2000` reference frame, the `x`-axis is defined to lie in the intersection of two planes: the plane of the Earth's equator and the plane of the Earth's orbit about the Sun. The `z`-axis is perpendicular to the Earth's equator. The `y`-axis completes a right-handed

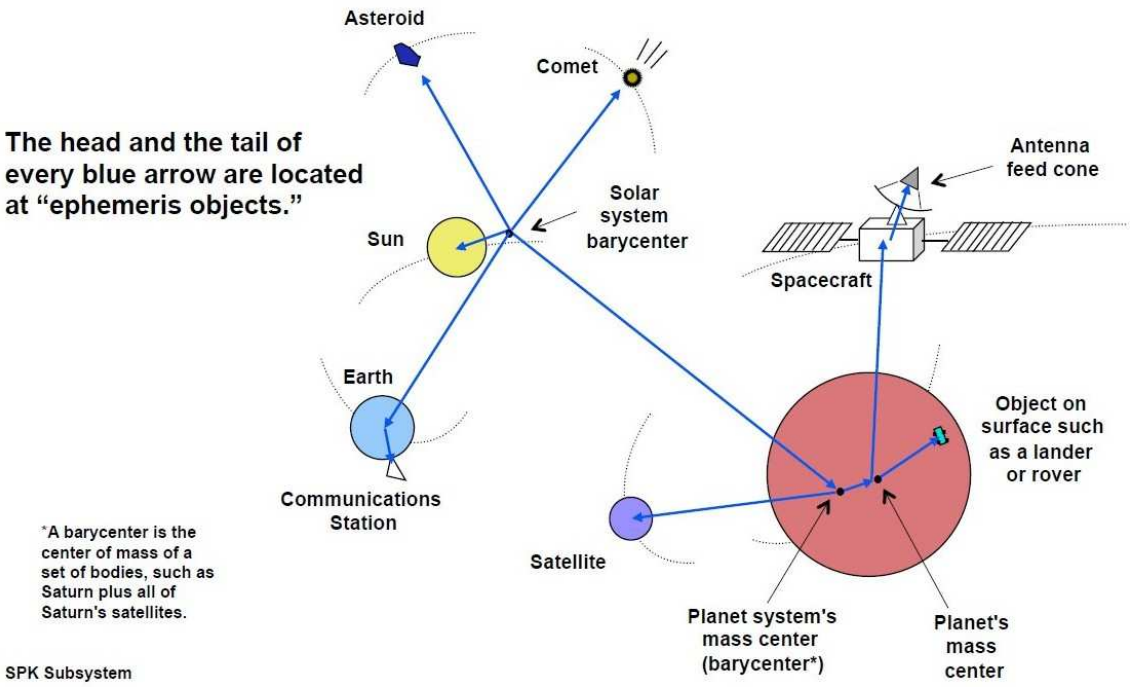

FIG. 1. Example of Ephemerides Objects [**12**].

system. The center of the frame is typically taken to be the solar system barycenter. (Note we are not attempting to rigorously define the `J2000` frame here. We are only illustrating how reference frames are defined. Many more details are required for a rigorous definition of the `J2000` frame [**9**]).

- State: A state is an array of six double precision numbers. The first three numbers give the `x`, `y`, and `z` coordinates respectively for the position of some object relative to another object in some Cartesian reference frame. The next three numbers give the velocity (`dx/dt`, `dy/dt` and `dz/dt` respectively) of the object with respect to the same reference frame.
- Inertial Frame: An inertial frame, is one in which Newton's laws of motion apply.
- Non-Inertial Frame: A non-inertial frame is a frame that rotates with respect to the celestial background. For example a frame whose axes are fixed with respect to the features on the surface of the Earth is a non-inertial frame.
- Ephemerides Time (ET): Ephemerides time, ET, is the independent variable in the equations of motion that describes the positions and velocities of objects in the solar system. In `SPICELIB` ET is used as a synonym for Barycentric Dynamical Time.

- Seconds Past 2000: In the SPK system times are specified as a count of seconds past a particular epoch (the epoch of the `J2000` reference frame). This reference epoch is within a second or two of the UTC epoch: 12:01:02.184 Jan 1, 2000 UTC. Epochs prior to this epoch are represented as negative numbers. The "units" of ET are designated in several different ways: seconds past 2000, seconds past J2000, seconds past the Julian year 2000, seconds past the epoch of the J2000 frame. All of these phrases mean the same thing.
- SPK segment: The trajectories of bodies in SPK files are represented in pieces called segments. A segment represents some arc of the full trajectory of an object. Each segment contains information that specifies the trajectory of a particular object relative to a particular center of motion in a fixed reference frame over some particular interval of time. From the point of view of the SPK system segments are the atomic portions of a trajectory.

On the other hand, SPICE system kernel files and subroutines refer to ephemerides objects, reference frames, and instruments by integer codes. In theory, a unique integer can be assigned to each body in the solar system, including interplanetary spacecraft. SPICE uses integer codes instead of names to refer to ephemerides bodies for three reasons:

- Space: Integer codes are smaller than alphanumeric names.
- Uniqueness: The names of some satellites conflict with the names of some asteroids and comets. Also, some satellites are commonly referred to by names other than those approved by the IAU.
- Context: The type of a body (barycenter, planet, satellite, comet, asteroid, or spacecraft) and the system to which it belongs (Earth, Mars, Jupiter, Saturn, Uranus, Neptune, or Pluto) can be recovered algorithmically from the integer code assigned to a body. This is not generally true for names.

The smallest positive codes are reserved for the Sun and planetary barycenters, `1,...,10` (As Mercury and Venus have no satellites, their barycenters (1 and 2) occupy the same locations as their mass centers (199 and 299)). Planets have ID codes of the form `P99`, where `P` is `1, ..., 9` (the planetary ID); a planet is always considered to be the `99th` satellite of its own barycenter, e.g. Jupiter is body number 599. Natural satellites have ID codes of the form `PNN` where `P` is `1,...,9` and `NN` is `01,...,98`, e.g. Europa, the `2th` satellite of Jupiter, is body number 502.

Moreover, the system reference of the Sun and planetary barycenters is refered to the Solar System Barycenter (ID 0). And the system reference of the planets and their satellites are refered at their planetary barycenters. As this diagram shows:

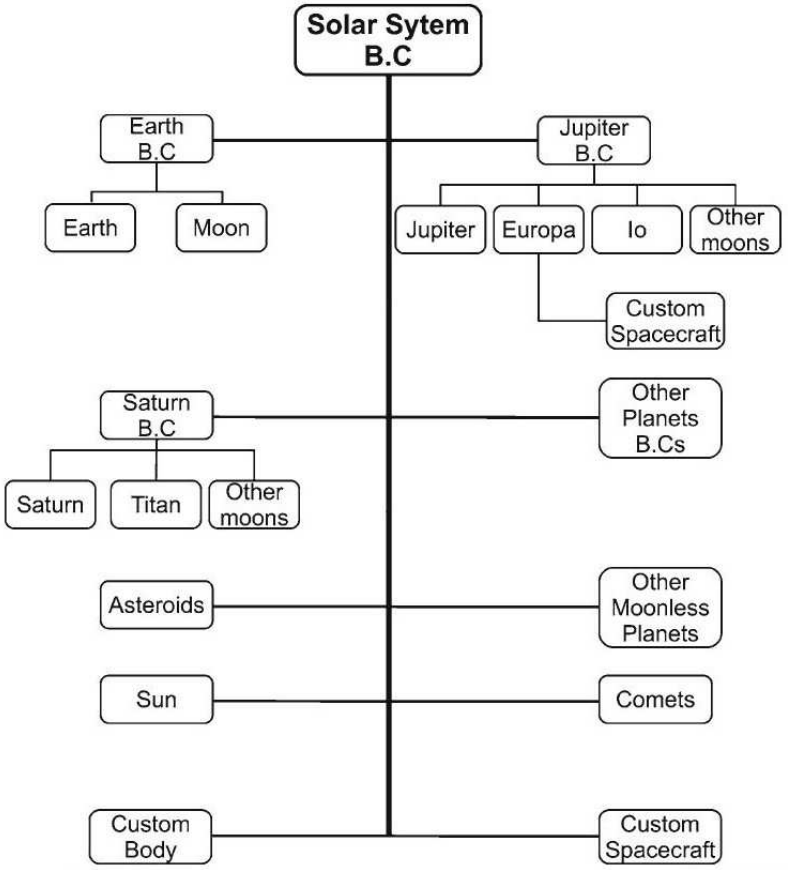

FIG. 2. Diagram of references [**1**].

1.1. SPK file structure. An SPK file is made up of one or more data ”segments” and a ”comment” area. These components are described below.

- Segments: An SPK file contains one or more ”segments”. Each segment contains enough ephemerides data to compute the geometric state (position and velocity) of one solar system body (the ”target”) with respect to another (the ”center”) at any epoch throughout some finite interval of time. The data in each segment are stored as an array of double precision numbers. The summary for the array, called a ”descriptor”, has two double precision components: 1) The initial epoch of the interval for which ephemerides data are contained in the segment, given in

ephemerides seconds past Julian year 2000; 2) The final epoch of the interval for which ephemerides data are contained in the segment, given in ephemerides seconds past Julian year 2000. The descriptor has six integer components: the NAIF integer code for the target; the NAIF integer code for the center, the NAIF integer code for the reference frame, the integer code for the representation (type of ephemerides data), the initial address of the array and the final address of the array. In addition to a descriptor, each array also has a name. The name of each array may contain up to 40 characters. This space may be used to store a brief description of the segment. For example, the name may contain pedigree information concerning the segment or may contain the name of the object whose position is recorded in the segment.

- Segment Order and Priority: Segment order implies priority. For a given target body, segment priority increases with distance from the start of the file: segments closer to the end of the file have higher priority than segments for the same target body that occur earlier in the file.
- The Comment Area: Preceding the ”segments”, the Comment Area provides space in the SPK file for storing textual information besides what is written in the array names. Ideally, each SPK file would contain internal documentation that describes the origin, recommended use, and any other pertinent information about the data in that file. For example, the beginning and ending epochs for the file, the names and NAIF integer codes of the bodies included, an accuracy estimate, the date the file was produced, and the names of the source files used in making the SPK file could be included in the Comment Area.
- SPK Data Types: The fourth integer component of the descriptor (the code for the representation, or ”data type”) is the key to the SPK format. There are several data types currently supported by `SPICELIB` software, but we are going to talk about two: Chebyshev polynomials (position only) and Chebyshev polynomials (position and velocity).
  - Chebyshev polynomials (position only): This SPK data type contains Chebyshev polynomial coefficients for the position of the body as a function of time. Usually, this data type is used for planet barycenters, and for satellites whose ephemerides are integrated. (The velocity of the body is obtained by differentiating the position). Each segment contains an arbitrary number of logical records. Each record contains a set of Chebyshev coefficients valid throughout an interval of fixed length.

– Chebyshev polynomials (position and velocity): This SPK data type contains Chebyshev polynomial coefficients for the position and velocity of the body as a function of time. Usually, this data type is used for satellites for which the ephemerides are computed from analytical theories. The structure of the segment is nearly identical to that of the SPK data of Chebyshev polynomials for position only. The only difference is that each logical record contains six sets of coefficients instead of three.

Chapter 3
Software package description

The main objetive of this software is to create a package to handle files of ephemerides and to manage the list of the bodies that we have selected, and, using this package, to build several programs to calculate change of coordinates, vector fields, etc.

This software package, programmed on Fortran 77 language, once the ephemerides file is read, allows to perform necessary calculations and integrations to do different works. To perform these calculations, library "`spicelib`" is used to access to the ephemerides corresponding files to planets and satellites (which are stored on BSP files), and with this information, calculate position, speed, acceleration and over-acceleration for a body with an ET (time) given previously.

With this information, and taking some constants (that have been saved previously in corresponding files), the software performs necessary calculations and integrations to obtain orbits for the given body, and carry out system reference changes.

1. Database implementation of the software

We have created this software to allow manage database of ephemerides as we want. The data of ephemerides have been extracted from the JPL ephemerides files. These files have the extension `*.bsp` and we have had to use the library `spicelib` to can extract all values that we need. In these files, the bodies for which we want all information, are identified uniquely by a number . This number takes the convention, as we have explained, the planetary barycenters and the Sun are identified by a number between 1 and 10. The Moon is identified by the ID 301, and the planets by the form `P99` where `P` is the ID of the barycenter appropriate. Finally, the natural satellites of the planets have the form `PNN` where `P` is `1,...,9` and `N` is `01,...,98`, ie.:

ID	NAME	ID	NAME
0	'SSB'	199	'MERCURY'
0	'SOLAR SYSTEM BARYCENTER'	299	'VENUS'
1	'MERCURY BARYCENTER'	399	'EARTH'
2	'VENUS BARYCENTER'	499	'MARS'
3	'EMB'	401	'PHOBOS'
3	'EARTH-MOON BARYCENTER'[1]	599	'JUPITER'
3	'EARTH BARYCENTER'	501	'IO'
4	'MARS BARYCENTER'	502	'EUROPA'
5	'JUPITER BARYCENTER'	699	'SATURN'
6	'SATURN BARYCENTER'	606	'TITAN'
7	'URANUS BARYCENTER'	799	'URANUS'
8	'NEPTUNE BARYCENTER'	701	'ARIEL'
9	'PLUTO BARYCENTER'	899	'NEPTUNE'
10	'SUN'	804	'THALASSA'
301	'MOON'	999	'PLUTO'

When we want to select different bodies to allow work with them, we need to open the file `modeleph.dat`. In this file is written the ID of the body, a number to indicate if we want work with this body (setting a "1" value in the file) or not (setting a "0" value), the name of the file that contains the body and the name of the body appropriate to the ID. For example, if we want to work with a model to contain Jupiter, Saturn, Earth, Moon, Sun and Io, the file `modeleph.dat` must be as follows:

```
1      3 1 DE406s        !Earth barycenter
2      5 1 DE406s        !Jupiter barycenter
3      6 1 DE406s        !Saturn barycenter
4      301 1 DE406s      !Moon
5      10 1 DE406s       !Sun
6      501 1 JUP230L     !Io
```

Must be taken in mind, that if we want work with a planet, his barycentre must be selected with a "1", and, if we want to work with a satellite, his planet and his barycentre must be selected. If this requirement is not satisfied, the program displays an error and

[1]To distinguish to Earth Barycenter, we have created the ID 12 for 'Earth-Moon Barycentre'.

stops. This is because the state of a planet or a satellite is referred to his barycentre, such that indicate the figure 2. But, if we really don't work with the barycentre when we do integrations, for example, we can use some subroutines to indicate this. It is what we will call like "activate or deactivate" some body.

Once we have selected all the bodies that we want, this information will be stored in a matrix that we will call `LIST`. This variable, as another, will be global to all software. The reason because we have global variables is that we will work with them always, and, due to memory and performance reasons, is better than local variables. When we finally have decided with what bodies we want work and if they must be activate or deactivate, all this information will be stored in `LIST`.

Now, we will explain, with more detail, the package files for the database and their construction. We will show brief fragments of code to illustrate what we are doing.

1.1. Common model. Fortran 77 has no global variables, i.e. variables that are shared among several program units (subroutines). The way to pass information between subroutines is to use the subroutine parameter list. Sometimes this is inconvenient, for example when many subroutines share a large set of parameters. In such cases one can use a common block. This is a way to specify that certain variables should be shared among certain subroutines.

We build a common block in the file "**commonmodel.inc**". This file shows the structures common to all the files. These variables are basic elements to work with the rest of subroutines. Next,we are going to explain the variables contained in "**commonmodel.inc**":

- **IMAX**: It is a parameter to indicate the maximum size for an array. In this case, it is set at `IMAX=53` because is the number of bodies we are working with.
- **ICO**: It contains the number of the body with the role of small primary, if it want to see orbits in a 'RTBP' system using the routines which change coordinates `(transs.f,transsplrtbp.f,...)`. For example, if it want a Sun-Barycenter Jupiter system, `ICO=5`, or Io-Jupiter system, `ICO=501` (501 id of Io).
- **IBC**: IBC shows the central body for the origin of coordinates. If `IBC=0`, is the usual JPL-ephemerides origin; if `IBC`$\neq$`0` it takes body `IBC` as origin of the coordinate frame.

- **XMUE(IMAX)**: It is a vector of `IMAX` columns. It indicates gravitational constants of all the bodies of the solar system in km^3/day^2. It has been computed from the values taken from the ephemerides files.
- **SEMIAX(IMAX)**: Vector of `IMAX` columns. They are the semiaxes of the orbits of the bodies of the solar system in km. It has been taken from the Explanatory Supplement ([**9**]) since they are not in the ephemerides files. It should be noted that the semiaxes of the Earth and the one of the Earth-Moon barycentre are both the same one.
- **XMU**: It is the mass parameter of the 'RTBP' considered (only if `ICO`$\neq$`0`).
- **ENEM**: It shows mean motion in km/day of a selected system. It has been computed using Kepler's thrid law: `XMUE(p)+XMUE(s)=(ENEM`^{2}`)*(SEMIAX(ICO)`^{3}`)`
- **LIST(3,IMAX)**: It is a array of three files and `IMAX` columns, it contains information about the model considered. The first row of `LIST` contains the identifier of the bodies that we are considering. The second row shows if we are considering position of the body (takes the value one); position and velocity (value two); position, velocity and acceleration (value three); or, position, velocity, acceleration and over-acceleration (value four). And the third row indicates if is a barycentre (value zero), a planet (value one) or a satellite (value two).
- **IHANDLE(IMAX)**: It is a vector of `IMAX` columns, it is a handle to access the ephemerides files.
- **ARXMO(IMAX)**: It is a vector to save the path of ephemerides files.

1.2. ***"modeleph.dat".*** In this file there are three columns. The first column is the identifier of all bodies with which we are working. The second is a zero if we are not considering the body or one if we are considering it. The third column is the file name where the body is. The last one is just a comment with the body name corresponding with the identifier.

The identifiers of the bodies will be saved in `LIST` in the order in which appear in this file.

```
1        1 1 DE406s   !Mercury barycenter DE406s
2        2 1 DE406s   !Venus barycenter
3        3 1 DE406s   !Earth barycenter
4        4 1 DE406s   !Mars barycenter
5        5 1 DE406s   !Jupiter barycenter
6        6 1 DE406s   !Saturn barycenter
```

```
7        7 1 DE406s   !Uranus barycenter
8        8 1 DE406s   !Neptune barycenter
9        9 1 DE406s   !Pluto barycenter
10       301 1 DE406s !Moon
11       10 1 DE406s  !Sun
12       3 1 DE406s   !Earth+Moon barycenter
13       199 0 DE406s !Mercury
14       299 0 DE406s !Venus
15       399 1 DE406s !Earth
16       401 0 MAR085 !Phobos MAR085
17       402 0 MAR085 !Deimos
18       499 0 MAR085 !Mars
19       501 1 JUP230L !Io   JUP230L
20       502 1 JUP230L !Europa
```

1.3. Subroutine *nmodjpl*. This file is the central module created to coordinate calls to other subroutines and read necessary input data of the file **"model.dat"**. Such as initialization of all the basic constants such as the mass ratios of the planets, semiaxes and the JPL-model of the solar system adopted.

This routine open the file "`model.dat`" to read it. First, it reads which origin we want, in this software we have implemented the origin in `J2000` that we have explained before. After this, it reads `IBC` to indicate where the origin of the coordinate frame is located (`IBC=0` usual JPL-ephemerides origin, `IBC`$\neq$`0` takes body `IBC` as origin of the coordinate frame). Below, it reads `ICO` to know if we are going to work with orbits in a 'RTBP' reference system or not (`ICO=0`). And finally, we read the number of the equilibrium point L_i, we usually use the Lagrange points L_1 (between the primaries) and L_2 (behind the small primary), if any.

After of this, we call several subroutines that we introduce here briefly. The subroutine **"read _modeleph"** reads the file "`modeleph.dat`" and takes the bodies that we are interested on. This subroutine creates a new file with only the bodies that we are considering. In `nmodjpl`, we read this new file and, from this, we call the handles of the files (*.bsp) of the bodies.

Then, there are subroutines that we can comment or uncomment depending on we are looking for. There are two subroutines, **"barorcom"** and **"dessat"** to activate or desactivate a body in the list. They does similar action as if the user had selected a

value ”1” to choose the body or a value ”0” to not choose the body in `modeleph.dat` file. Subroutine **barorcom** receives two values as parameters. One of them is the identifier of the body and second one is the mode that should be applied (”0” deactivate all (satellite,planet and barycenter), ”1” barycentre activated, ”2” planet and satellite activated). Subroutine **dessat** only receives one value as parameter, that corresponds with identifier of the satellite that we want to include or exclude from the model.

Also, we write this list in a file (`list.dat`) with the routine **”writelist”**, to can see the bodies selected.

Below, we have four routines to build constants that we will need. And various routines to work with the list (ordering the list and inserting or removing an element of it).

Finally, we have built main programs to propagate orbits from initial conditions and to change coordinate systems.

```
1        SUBROUTINE NMODJPL
2        ..............
3        C Write LIST in file "list.dat"
4        OPEN(IUNIT,FILE='list.dat')
5        CALL WRITELIST(IUNIT,'list.dat')
6        CLOSE(IUNIT)
7        c Fill the GM values of bodies
8        CALL FILLXMUE(XMUETOT)
9        c Fill semiaxes of bodies
10       CALL FILLSEMIAXES(SEMIAXTOT)
11       C Take the values of GM and semiaxes corresponding to the selected
             bodies
12       DO I=1,LFIN
13         XMUE(J)=XMUETOT(IPOS(I))
14         SEMIAX(J)=SEMIAXTOT(IPOS(I))
15         J=J+1
16       END DO
17       c If no RTBP is chosen, the task of this routine is completed
18       IF (ICO.EQ.0) GOTO 15
19       c Computing XMU and ENEM
20       CALL XMU_ENEM(XMUETOT,SEMIAXTOT)
21       c Computing the scaling factor gamma
22       CALL FACTGAMMA
23       c Check if the LIST is ordered. If it isn't ordered, it orders
24       CALL ORDER
25       ..............
```

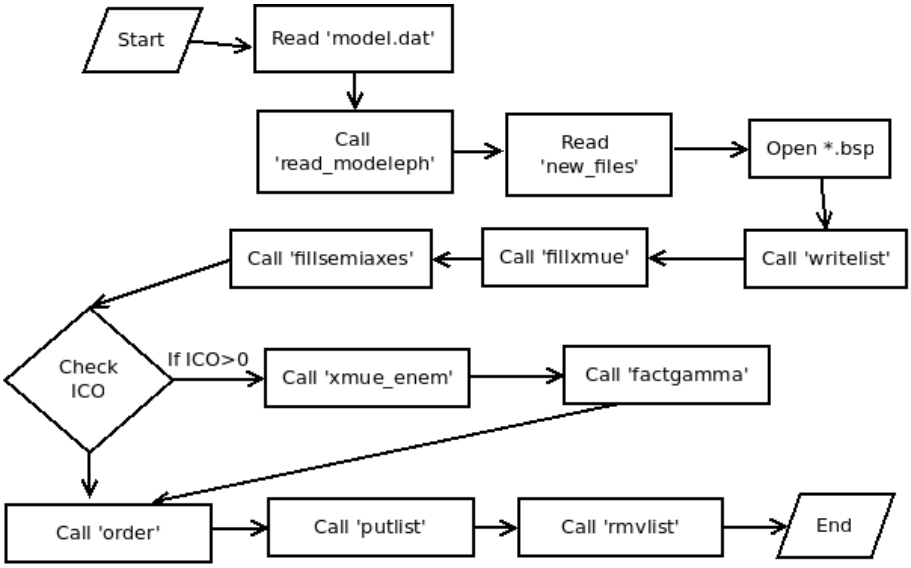

FIG. 1. Workflow diagram of nmodjpl.

1.4. Subroutine *read_modeleph*. We extract data of the file `"modeleph.dat"`. The file has a matrix data structure of 53 rows and 3 columns. For each row (that identifies an object as a planet barycentre, a planet or a satellite), we have 3 columns that contains identifier of the body (first column), flag to indicate if the body is selected or not (second column), and the file in which the body's data is stored (third column).

Must be checked the value of the body `ICO`, and, if it is not zero, if the corresponding body with this value was considered. Following this, it calls to the subroutine **"change_file_dat.f"** to create a new file with only the bodies taken.

It checks if the final list is congruent with **"cong.f"**, i.e., if we take a planet, the barycenter appropriate must be taked, and if we take a satellite, the planet and the barycenter correspondent should be taked also.

Finally, we fill the common variable `LIST`. In the first row of this array, we introduce the identifiers of the selected bodies. In the second row, for now, we just indicate that we want positions of bodies putting "1". And, in the third row, we mark if it is a barycenter, a planet or a satellite.

```
1       SUBROUTINE READ_MODELEPH(IPOS)
2       ..............
3       C Open data file
4       CALL GETLUN(IUNIT)
```

```
      ARXJ='modeleph.dat'
      OPEN(IUNIT,FILE=ARXJ,STATUS='OLD')
      C Read data from data file
      I=1
      DO WHILE (.TRUE.)
        READ(IUNIT,*, END=999) IDENTIF(I),DISP(I), ARCH_CARG(I)
        I=I+1
      END DO
      999     CONTINUE
      ........
      C Create a new .DAT with the new values
      CALL CHANGE_FILE_DAT(ARCH_CARG2)
      CLOSE(IUNIT)
      C See if the list is congruent
      CALL CONG(NUM_LISTA)
      ..............
```

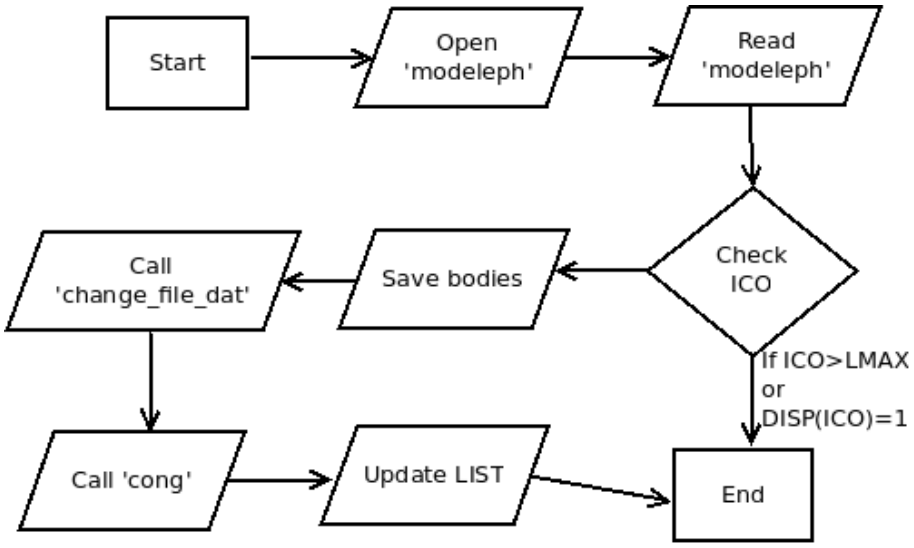

FIG. 2. Workflow diagram of leer_modeleph.

1.5. Subroutine *plajplnsat*. It obtains states of a body (position, velocity, acceleration and over-acceleration), using functions of the **spicelib** library. Given an epoch and a body, we extract the data from the appropriate ephemerides file to calculate the state of the body. Should be noted that the time in subroutine `plajplnsat` comes in Modified Julian Dates, and in order to obtaint the states we will be using time in seconds.

With the time and the chosen body, it does the extraction of data with the function **"spkg"**, that we explain below. Finally, the states that we obtain are corverted to km/day, km/day^2 and km/day^3.

In subroutine **"spkg"** to calculate states of the body, we use functions of the **spicelib** library. Function **"spksfs"** is called, this function searches through loaded files to find the first segment applicable to the body and epoch specified. Buffer searched segments in the process, to attempt to avoid re-reading files. This routine finds the highest-priority segment, in any loaded `SPK` file, such that the segment provides data for the specified body and epoch. Also, subroutines **"spkr02"** and **"spkr03"** are called. Routine `"spkr02"` reads a single SPK data record from a segment of type 2 (Chebyshev, position only), and routine `"spkr03"` reads a single SPK data record from a segment of type 3 (Chebyshev coefficients, position and velocity).

Finally, if we need acceleration and over-acceleration, we have built subroutine **"spke0203ipv"**.

```
      SUBROUTINE PLAJPLNSAT(ET,IPVAS,PVAS)
      ...................
      c Time in seconds
      SECS=(ET-0.5D0)*DAYSEC
      c Calculating position, velocity, acceleration and overacceleration
      ICONT=1
      DO I=1,LFIN
        IF (LIST(2,I).NE.0) THEN
        DO K=1,12
          TSTATE(K)=0
        END DO
        IAUX_HNDL=IHANDLE(ICONT)
        J=LIST(1,I)
        IF (IPVAS.EQ.0) THEN
          IPV=LIST(2,I)
        ELSE
          IPV=IPVAS
        END IF
        CALL SPKG(IPV, J, SECS, IAUX_HNDL, TSTATE)
      ...................
```

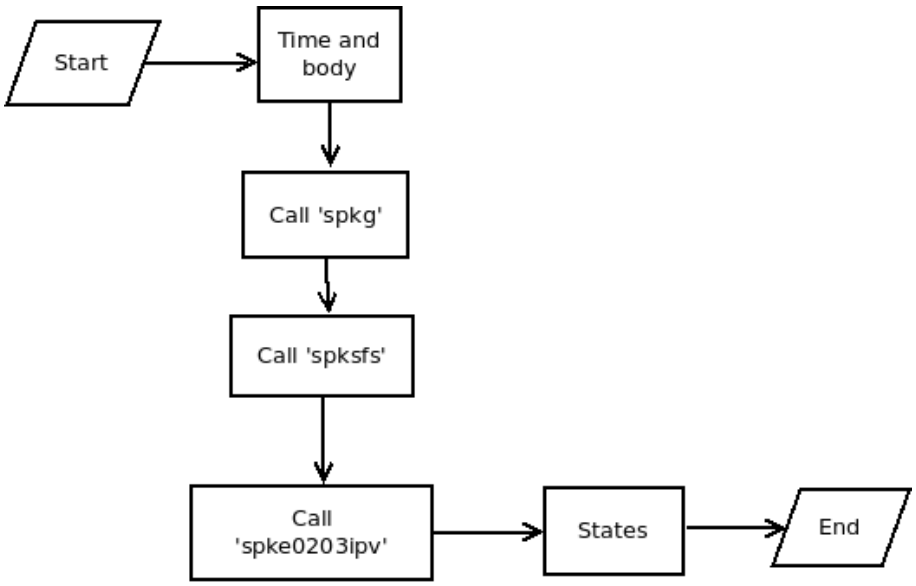

FIG. 3. Workflow diagram of plajplnsat.

2. Vector fields and changes of coordinates

In this section we will explain the vector fields and the changes of coordinates that we have used to implement our software. First, we will show the vector field that we have implemented for the rectricted problems with the change of coordinates appropriate. After this, the vector fields for sideral and synodic coordinates, with their changes of coordinates.

Finally, we will explain, in general, how we have built the program for this, and then, more specifically, we will show the subroutines that we have done for this purpose.

2.1. Solar System model. Most restricted problems take as starting point the Restricted Three Body Problem (RTBP). We recall that it models the motion of a particle under the gravitational attraction of two primaries which are assumed to be point masses revolving in circular orbits around their center of mass. The Hamilton function of this system is, in a coordinate system that revolves with the primaries (such a system is called synodic),

$$H(x,y,z,p_x,p_y,p_z) = \frac{1}{2}(p_x^2+p_y^2+p_z^2) + yp_x - xp_y - \frac{1-\mu}{[(x-\mu)^2+y^2+z^2]^{1/2}} - \frac{\mu}{[(x-\mu+1)^2+y^2+z^2]^{1/2}}$$

being $\mu = m_2/(m_1 + m_2)$, where $m_1 > m_2$ are the masses of the primaries.

We will consider, instead of taking as starting equations those of the RTBP, Newton's equation for the motion of an infinitesimal body in the force field created by the bodies of the Solar System

$$R'' = G \sum_i m_i \frac{(R_i - R}{\|R - R_i\|^3}, \tag{7}$$

where G is the gravitational constant, R is the position of the infinitesimal body, $\mathtt{R}_i$ is the position of the Solar System body i in some suitable inertial system and $\mathtt{m}_i$ its mass.

Below, we introduce suitable reference systems and units such that, after selecting two bodies of the Solar System as primaries, the above equations are set as a perturbation of the RTBP. We will denote the set of bodies of the Solar System by $\mathtt{S}=\mathtt{P}_1,\ldots,\mathtt{P}_{11}$ where $\mathtt{P}_1,\ldots,\mathtt{P}_{11}$ are the nine planets, the Moon and the Sun, respectively. The mass of $\mathtt{P} \in \mathtt{S}$ will be denoted by $\mathtt{m}_p$. It should be noted that, althought in this text we will talk only about $\mathtt{P}_1,\ldots,\mathtt{P}_{11}$, in our software, this model is extended to the natural satellites of the planets.

In an inertial reference system, the Lagrangian related to Newton's equations of motion 7 of an infinitesimal body Q under the gravitational action of the bodies in S, is

$$L(R, R', t^*) = \frac{1}{2}\langle R', R'\rangle + \sum_{i \in S} \frac{Gm_i}{k\|R - R_i\|},$$

where $\mathtt{R}=(\mathtt{X},\mathtt{Y},\mathtt{Z})^T$ is the position of Q, the prime denotes the derivative with respect to time, $\mathtt{t}^*$, $\langle R', R'\rangle$ is the dot product between R' and R', G is the gravitational constant, $\mathtt{R}_i$ is the position of the body $\mathtt{i} \in \mathtt{S}$, and $\|\cdot\|$ denotes the Euclidean norm. In practice, it is convenient that the reference frame and units, both in space and time, are consistent with the ephemerides data files used for the determination of $\mathtt{R}_i$.

Since we are interested in writing the equations of motion for Q as a perturbation of the RTBP equations, we must select two bodies $\mathtt{I},\mathtt{J} \in \mathtt{S}$ with $m_I > m_J$, which will play the role of primaries. In this way, the mass parameter, μ, is defined as $\mu = m_J/(m_I + m_J)$, and so $1 - \mu = m_I/(m_I + m_J)$. Next, we must introduce the

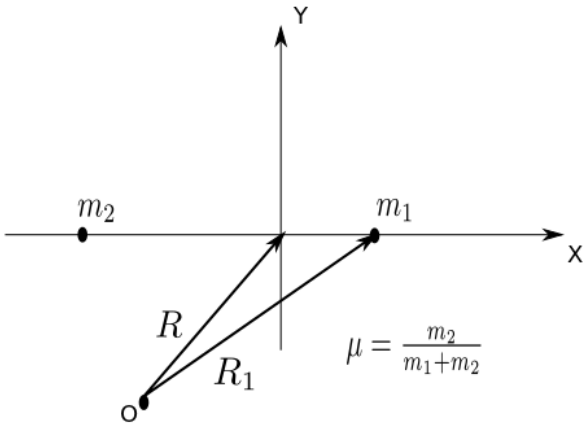

FIG. 4. Position of bodies.

synodic reference frame. Recall that the origin of this system is set at the barycenter of I, J and that the positions of the primaries are fixed at $(\mu, 0, 0)$ and $(\mu - 1, 0, 0)$.

The transformation from synodical coordinates, $\mathtt{r}=(\mathtt{x},\mathtt{y},\mathtt{z})^T$, to sideral ones, $\mathtt{R}$, is defined by

$$R = B + kCr, \tag{8}$$

where

- The translation, $\mathtt{B}$, is given by
  $B = \frac{m_I R_I + m_J R_J}{m_I + m_J}$,
  that clearly puts the barycenter of the primaries at the origin.
- The orthogonal matrix $\mathtt{C}=(\mathtt{e}_1,\mathtt{e}_2,\mathtt{e}_3)$, sets the primaries on the $\mathtt{x}$-axis and turns the instantaneous plane of motion of the primaries into the $\mathtt{xy}$ plane (by requiring that the relative velocity of one primary with respect to the other has its third component equal to zero). The columns of $\mathtt{C}$ are:
  $e_1 = \frac{R_{JI}}{\|R_{JI}\|}$, $e_3 = \frac{R_{JI} \times R'_{JI}}{\|R_{JI} \times R'_{JI}\|}$, $e_2 = e_3 \times e_1$
  being $\mathtt{R}_{JI}=\mathtt{R}_J-\mathtt{R}_I$.
- $\mathtt{k}=\|R_{JI}\|$ is a scaling factor which makes the distance between the primaries to be constant and equal to 1.

It is important to remark that this change of variables is non-autonomous, since $\mathtt{B}$,$\mathtt{k}$ and $\mathtt{C}$ depend on time through the components of $\mathtt{R}_I$ and $\mathtt{R}_J$.

The change of coordinates given by equation 8 is checked to preserve the Lagrangian form of the equations. Then, as R'=B'+k's+ks',

$$\langle R', R'\rangle = \langle B', B'\rangle + 2k'\langle B', s\rangle + 2k\langle B', s'\rangle + k'^2\langle r, r\rangle + 2kk'\langle s, s'\rangle + k^2\langle s', s'\rangle,$$

and for a body i, R_i=B+kCr_i, and C is orthogonal, then

$$\|R - R_i\| = \|kCr - kCr_i\| = k\|r - r_i\|.$$

The new Lagrangian becomes

$$L(r, r', t^*) = \frac{1}{2}\langle B', B'\rangle + k'\langle B', s\rangle + k\langle B', s'\rangle + \frac{1}{2}k'^2\langle r, r\rangle + kk'\langle s, s'\rangle + \frac{1}{2}k^2\langle s', s'\rangle + \frac{Gm_I}{k[(x-\mu)^2 + y^2 + z^2]^{1/2}} + \frac{Gm_J}{k[(x-\mu+1)^2 + y^2 + z^2]^{1/2}} + \sum_{i\in S^*}\frac{Gm_i}{k\|r - r_i\|},$$

where s=Cr, r_i is the position of the body i in dimensionless coordinates and S^* represents the set of Solar System bodies without the two primaries I,J. To get above expression of L, we use that C defines an orthogonal transformation and, hence, it preserves the scalar product and the Euclidean norm.

Finally, we want to use the same time units as those usual for the RTBP, where 2π time units correspond to one revolution of the primaries. If t^* is some dynamical time and n is the mean motion of J with respect to I, then we perform the change of independent variable throught

$$t = n(t^* - t_0^*),$$

where t_0^* is a fixed epoch, t will be called dimensionlees time. We should remark that, in our program, when we take Earth+Moon it means the Earth-Moon barycenter and, for this system, the Earth and the Moon are substituted in S by a fictitious body of mass m_E+m_M behaving as their barycenter. Using Kepler's third law, $G(m_I + m_J) = n^2a^3$, we can also define the mean semi-major axis of the orbit of one primary around the other.

If we denote with a dot the derivative with respect to t, then the new Lagrangian can be written as

$$L(r, \dot r, t) = n^2\left(\frac{1}{2}\langle \dot B, \dot B\rangle + \dot k\langle \dot B, s\rangle + k\langle \dot B, \dot s\rangle + \frac{1}{2}\dot k^2\langle r, r\rangle + k\dot k\langle s, \dot s\rangle + \frac{1}{2}k^2\langle \dot s, \dot s\rangle\right) + \frac{Gm_I}{k[(x-\mu)^2 + y^2 + z^2]^{1/2}} + \frac{Gm_J}{k[(x-\mu+1)^2 + y^2 + z^2]^{1/2}} + \sum_{i\in S^*}\frac{Gm_i}{k\|r - r_i\|}.$$

From this Lagrangian we can remove the term $\langle \dot{B}, \dot{B} \rangle$, since it is independent of `r` and $\dot{r}$, and multiply by the scaling factor $a/(G(m_I + m_J)) = 1/(n^2 a^2)$ without affecting the equations of motion. In this way we get

$$L(r, \dot{r}, t) = \frac{1}{a^2}\left(\dot{k}\langle \dot{B}, s\rangle + k\langle \dot{B}, \dot{s}\rangle + \frac{1}{2}\dot{k}^2\langle r, r\rangle + k\dot{k}\langle s, \dot{s}\rangle + \frac{1}{2}k^2\langle \dot{s}, \dot{s}\rangle\right)$$
$$+ \frac{a}{k}\left(\frac{1-\mu}{[(x-\mu)^2 + y^2 + z^2]^{1/2}} + \frac{\mu}{[(x-\mu+1)^2 + y^2 + z^2]^{1/2}} + \sum_{i \in S^*} \frac{\mu}{k\|r - r_i\|}\right).$$

where $\mu_i = m_i/(m_I + m_J)$.

Since `e1, e2, e3` form an orthogonal basis, we have that $\langle e_i, e_j\rangle = \delta_{ij}, \langle \dot{e}_i, e_j\rangle = -\langle e_i, \dot{e}_j\rangle$ and $\langle \dot{e}_i, e_i\rangle = 0$ for `i, j=1, 2, 3`. It can be further shown that $\langle \dot{e}_1, \dot{e}_2\rangle = 0, \langle \dot{e}_2, \dot{e}_3\rangle = 0$ and $\langle \dot{e}_1, e_3\rangle = 0$.

2.2. Model in sideral and synodic coordinates. Here, when we refer to the real Earth-Moon system, we mean the gravitational model of the nine planets, the Moon and the Sun. But, in practice, we will extend the model for all bodies (planets and natural satellites) of the Solar System.

Their positions are given by the JPL-ephemerides. Denote the set of these bodies as `S=` $(\mu_1, ..., \mu_{11})$. where $\mu_1, ..., \mu_9$ indicate the reduced masses of the nine planets and μ_{10}, μ_{11} indicate the reduced masses of the Moon and the Sun ($\mu_i = m_i/(m_3 + m_{10})$, where m_i (`i = 1, ..., 11`) are the masses of the nine planets, the Moon and the Sun). The mass unit is the sum of the masses of the Earth and the Moon. In an Earth-centered sideral coordinate, the small body is defined as follows

$$\ddot{\vec{R}} = -\mu_3 \frac{\vec{R}}{R^3} - \sum_{i \in S, i \neq 3} \mu_i \left(\frac{\vec{\Delta}_i}{\Delta_i^3} + \frac{\vec{R}_i}{R_i^3}\right) \tag{9}$$

where $\vec{R}$ is the position vector of the small body, $\vec{R}_i$ is the position vector of the major body μ_i, and $\vec{\Delta}_i$ is the position vector of the small body from the major body μ_i.

Denoting the corresponding variables in the Earth-centered synodic coordinate as $\vec{r}, \vec{r}_i, \vec{\delta}_i$, the following relations hold (similar relations hold for $\vec{r}_i, \vec{\delta}_i$).

Now, the transformation from synodic coordinates, `r`, to sideral ones, `R`, will be:

$$R = Cr, \dot{R} = C\dot{r} + \dot{C}r, \ddot{R} = \ddot{C}r + 2\dot{C}\dot{r} + C\ddot{r} \tag{10}$$

where C is the rotation matrix, determined by the motion of the Moon with respect to the Earth

$$C = (e_1, e_2, e_3); e_1 = \frac{R_{10}}{k}, e_3 = \frac{R_{10} \times \dot{R}_{10}}{\|R_{10} \times \dot{R}_{10}\|}, e_2 = e_3 \times e_1$$

where k=$|R_{10}|$. In our program, we extend this to all bodies. Keep in mind that the transformation from synodical to sideral coordinates is different from that given previously. There, the coordinate r is scaled by the instantaneous distance k between the Earth and the Moon. Here we keep the length unit fixed. Substituting equation 10 into 9 we obtain

$$\ddot{\vec{r}} = -2C^T\dot{C}\dot{\vec{r}} - C^T\ddot{\vec{r}} - \mu_3\frac{\vec{r}}{r^3} - \sum_{i\in S, i\neq 3}\mu_i\left(\frac{\vec{\delta_i}}{\delta_i^3} + \frac{\vec{r_i}}{r_i^3}\right).$$

2.3. Vector fields and changes of coordinates implementation. Another objetive of this software is to do integrations around a body. For this purpose, we have built the more common vector fields for the Restricted Three Body Problem (RTBP) and for some reference systems, such that inertial and synodic systems. Also, to have different possibilities, we have built several changes of coordinates, to work in coordinates RTBP, JPL, inertial or synodic.

To construct vector fields, we use the bodies that we have selected before, through the variable LIST. With these bodies, we choose the initial coordinates and as many states as we want (ie. if we only want position, position+velocity, position+velocity+acceleration or position+velocity+acceleration+overacceleration), and, for a given epoch, we calculate the vector field chosen. Also, for some vector field, we have implemented the variational equations. For example, if we want to do integrations around the Earth, using the vector field of the RTBP, we can select the Sun (for example) and we can configure the program to use variational equations. Should be taken in mind that, when we work in RTBP system, Earth and Sun (or other bodies) will be in the X axis.

Also, when, for example, we have calculated a vector field in determinate coordinates and we want to see this vector field in another coordinates, or simply, when we have the state of a body (position, position+velocity,...) in a coordinate system and we want to change to another, this software allows to perform this change of coordinates. For example, we can do change of coordinates between RTBP and JPL coordinates, in both directions, only indicating with a parameter of the routine selected, what is the direction

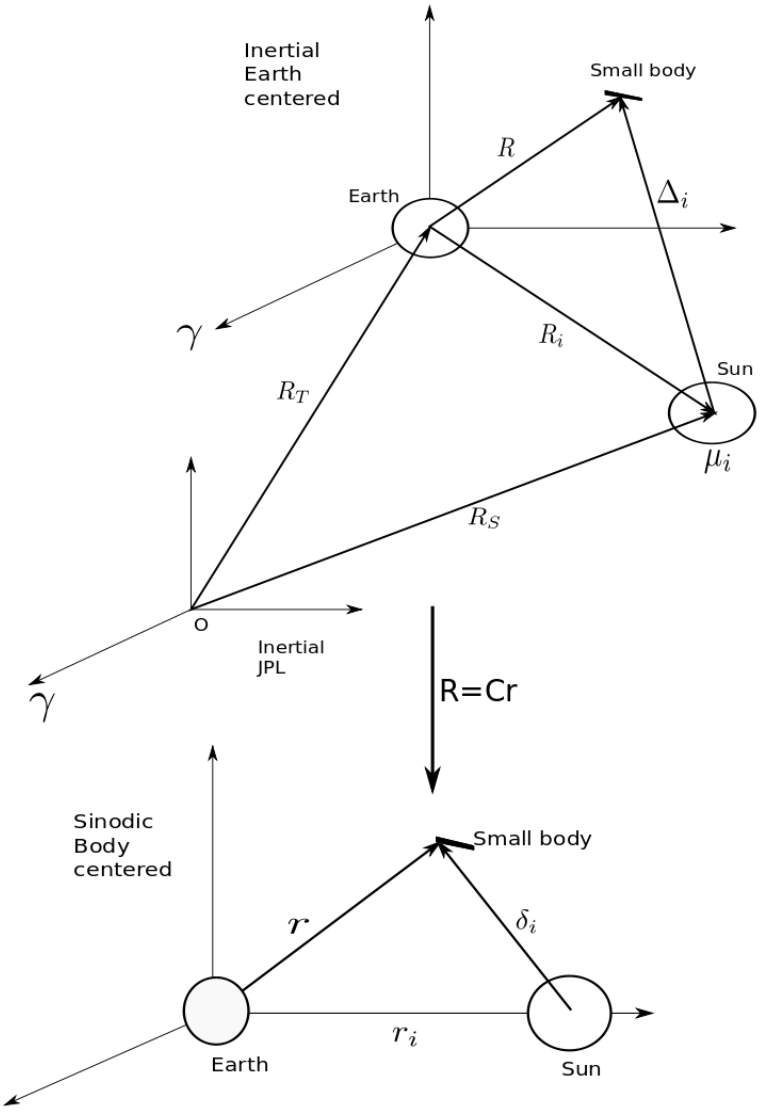

FIG. 5. Change of coordinates.

(ie., in the right variable we will set to ”1” if we want the change from JPL to RTBP, and to ”-1” if we want the inverse change).

With all these routines, it be able to do refinements in the integrations, to calculate orbits with more precision, such that we will show in the application of the software with the parallel shooting.

Below, we will explain, with more detail, the package files and their construction. We will show brief fragments of code to illustrate what we are doing.

2.4. Subroutines of vector fields. The subroutines **"derivcs","dervin","dervsin"** have been built to propagate integrations of ephemerides data. Using a main program for example, it is necessary to introduce initial conditions and from this, we call subroutine **"rk78"** for propagation. This last routine is an implementation of a Runge-Kutta-Fehlberg method of orders 7 and 8. Using a total of thirteen steps (and evaluations of the vector field) it computes two different estimations of the next point. The difference between both estimations (with local errors of order 8 and 9) is computed and the L_1 norm is obtained. This norm is divided by the number of equations. The value obtained in this way is required to be less than a given tolerance E_1 times $(1 + 0.01 * DD)$ where `DD` is the L_1 norm of the point computed to order 8. If this requirement is satisfied, the order 8 estimation is taken as the next point. If not, a suitable value of the step `H` is obtained and the computation is started again. In any case, when the next point is computed, a prediction of the step `H`, to be used in the next call of the routine, is done.

Routine `"rk78"` calls the subroutine **"derivcs"**, **"dervin"** or **"dervsin"** that we will explain below. These subroutines compute the vector field defining the equations of motion and the variational equations with respect to time and coordinates (if wanted) for a particle in the solar system according to a given model centered in some place of the solar system (by default, in the solar system barycenter). The result is an array consisting of components of the vector field in JPL-Equatorial coordinates with origin according to `IBC`. In the next section we will explain how we have made these vector fields.

It must observe that, in the list of bodies that we will want use, it is necessary include always the Sun.

```
      SUBROUTINE DERIVCS(A,B,N,F)
.....
      CALL PLAJPLNSAT(A,4,PVAS) !IPV=2
.....
      DO 2 I=1,LFIN
        IF (LIST(3,I).EQ.0) GOTO 2 !Barycentres are not taken
        DO 18 L=1,3
        X(L)=B(L)-PVAS(L,I)
        IF (N.GT.6) V(L)=PVAS(L+3,I)
18      CONTINUE
        R2=X(1)*X(1)+X(2)*X(2)+X(3)*X(3)
        R3=R2*DSQRT(R2)
        XMAS=XMUE(I)*DAYSEC
        XMAS=XMAS*DAYSEC
        CO=XMAS/R3
```

```
          DO 4 L=1,3
          F(L+3)=F(L+3)-CO*X(L)
    4       CONTINUE
          IF (N.EQ.6) GOTO 2
          AUX=0.D0
          DO 6 L=1,3
    6       AUX=AUX+X(L)*V(L)
          DO 7 L=1,3
          G(L)=G(L)+CO*(V(L)-3*X(L)*AUX/R2)
          DO 7 K=1,3
          IF(K.EQ.L) H(L,K)=H(L,K)-CO
    7       H(L,K)=H(L,K)+CO*(3*X(L)*X(K)/R2)
    2     CONTINUE
        IF (N.EQ.6) RETURN
        DO 9 J=1,6
        DO 9 I=1,3
        ............
        DO 8 K=1,3
        F(I+6*J+9)=F(I+6*J+9)+H(I,K)*B(K+6*J+6)
        F(I+6*J+27)=F(I+6*J+27)+H(I,K)*B(K+6*J+24)
    8     CONTINUE
```

```
      SUBROUTINE DERVIN(A,B,N,F)
      DO 118 I=1,3
        RT(I)=PVAS(I,IPOS_ICO)
  118   CONTINUE

      DO 120 K=1,LFIN
      IF ((K.NE.IPOS_ICO).AND.(LIST(3,K).NE.0)) THEN
      DO 119 L=1,3
        RS(L)=PVAS(L,K)
        RI(L)=RS(L)-RT(L)
        DELTAG(L)=B(L)-RI(L)
  119     CONTINUE
        SUMDELG=DSQRT(DELTAG(1)**2+DELTAG(2)**2+DELTAG(3)**2)
        SUMRIG=DSQRT(RI(1)**2+RI(2)**2+RI(3)**2)
        XMAS=XMUE(K)
  c XMUE in km**3/day**2
        XMAS=XMAS*DAYSEC
        XMAS=XMAS*DAYSEC
      DO I=1,3
        DELG(I)=DELTAG(I)/(SUMDELG**3)
        RIVECT(I)=RI(I)/(SUMRIG**3)
        SUMAG(I)=SUMAG(I)+XMAS*(DELG(I)+RIVECT(I))
      END DO
      END IF
```

```
25      120       CONTINUE
26
27        SUME=DSQRT(B(1)**2+B(2)**2+B(3)**2)
28        IF (SUME.EQ.0) SUME=1
29        XMAS=XMUE(IPOS_ICO)
30      c XMUE in km**3/day**2
31        XMAS=XMAS*DAYSEC
32        XMAS=XMAS*DAYSEC
33        DO I=1,3
34        F(I)=B(I+3)
35        F(I+3)=-XMAS*(B(I)/(SUME**3))-SUMAG(I)
36        END DO
37      ............
```

2.5. Subroutines of change of coordinates. These routines have been built to perform changes of coordinates between different reference systems. For this, and according to the change we want, we will call the subroutines **"transs"**, **"transsjplpl"**, **"transsplrtbp"** or **"transsinsin"**.

Routine **"transs"** does the transformation from adimensional 'RTBP' coordinates to JPL coordinates (`J2000` equatorial) or viceversa, according to the formula:

$$(EQ) = K * (C) * (AD) + B$$

where `(EQ)` is a vector with equatorial JPL coordinates, `(AD)` is another vector with adimensional RTBP coordinates, `K` is the scaling factor, `(C)` is a orthogonal matrix and `B` is the traslation (the construction of their columns depend on which reference system we have selected, we will explain this in more detail in the section "Vector fields").

In the 'RTBP' system, the big primary is located at $(\mu, 0, 0)$ with mass $1 - \mu$ and the small one at $(\mu - 1, 0, 0)$ with mass μ.

Routine **"transsk1"** is as `"transs"`, but the scaling factor `K` is set to $k = 1$. (And the same rule applies with all subroutines ending the name with *"K1"*.)

Routine **"transsplrtbp"** uses the same formula than `"transs"`, but this routine does the transformation from adimensional 'RTBP' coordinates to JPL coordinates centered in a planet barycenter (`IBC`) , or viceversa.

Routine **"transsjplpl"** makes the transformation from 'JPL' coordinates centered in SSB (Solar System Barycentre) to 'JPL' coordinates centered in the barycenter of a planet (IBC) or viceversa. This routine uses the subroutines `"transs"` and `"transsplrtbp"` to do the change.

Routine **"transsinsin"** transforms from inertial (`EQ`) to sinodical (`AD`) coordinates or viceversa, according to the formula:

$$(EQ) = (C) * (AD)$$

where `(C)` is orthogonal matrix.

These routines use the routine `"plajplnsat"` to calculate the states of the bodies.

Routine **"transli"** requires special mention. This routine transforms from adimensional barycentric coordinates in a 'RTBP' frame to addimensional-normalized (+ or - according to Richardson) coordinates with origin at an equilibrium point or viceversa. The reference frames at L_I are the following:

- L_1, L_2 and L_3:
  - `X` axis positive direction pointing oposite to the big primary (this is, for L_3 is the same sense as 'RTBP' and for L_1 and L_2 is oposite sense).
  - `Y` axis rotated 90 deg counterclockwise from `X` axis.
  - `Z` axis perpendicular to the plane pointing north.
- L_4 and L_5: the reference frame is the translation of the 'RTBP' one to the equilibrium point. The positive directions of the axis are not changed.

Let `X,Y,Z` coordinates of the 'RTBP' and `x,y,z` adimensional L_I, the explicit change of coordinates are (in all the cases the `z`-equation is `z`=$\gamma\cdot$ `Z` and for L_4 and L_5 $\gamma = 1$):

- $X = -\gamma * x + \mu - 1 + \gamma$
  $Y = -\gamma * y$ for L_1,
- $X = -\gamma * x + \mu - 1 - \gamma$
  $Y = -\gamma * y$ for L_2,

- $X = \gamma * x + \mu + \gamma$
  $Y = \gamma * y$ for L_3,
- $X = x - \mu + 0.5$
  $Y = y + \sqrt{3}/2$ for L_4,
- $X = x - \mu + 0.5$
  $Y = y - \sqrt{3}/2$ for L_5,

where γ is the adimensional position of the equilibrium point L_I, distance in 'RTBP' units from the point L_I to the nearest primary.

```
         SUBROUTINE TRANSS(DAY,EQ,AD,IPVA,IS)
.................
C First, we check if is a barycenter, the Moon, Sun, a planet or a satellite
36       IF (((IBC.NE.K).AND.(IBC.NE.ICO)).AND.(LIST(3,IPOS_ICO).EQ.0))
    .         THEN !Barycenter(primario menor)-Sun(p. mayor)
           DO 5 I=1,3
           E(I)=PVAS(I,IPOS_ICO)-PVAS(I,IPOS_K)
           XT(I)=PVAS(I,IPOS_K)+E(I)*XMU
           EP(I)=PVAS(I+3,IPOS_ICO)-PVAS(I+3,IPOS_K)
           IF (IPVA.EQ.1) GOTO 5
           XT(I+3)=PVAS(I+3,IPOS_K)+EP(I)*XMU
           EPP(I)=PVAS(I+6,IPOS_ICO)-PVAS(I+6,IPOS_K)
           IF (IPVA.EQ.2) GOTO 5
           XT(I+6)=PVAS(I+6,IPOS_K)+EPP(I)*XMU
           EPPP(I)=PVAS(I+9,IPOS_ICO)-PVAS(I+9,IPOS_K)
5          CONTINUE
         ELSE
           IF ((IBC.EQ.K)) THEN  !Earth-Moon
           WRITE(*,*) 'PUES NO'
           DO 6 I=1,3
           E(I)=PVAS(I,IPOS_ICO)
           XT(I)=E(I)*XMU
           EP(I)=PVAS(I+3,IPOS_ICO)
           IF (IPVA.EQ.1) GOTO 6
           XT(I+3)=EP(I)*XMU
           EPP(I)=PVAS(I+6,IPOS_ICO)
           IF (IPVA.EQ.2) GOTO 6
           XT(I+6)=EPP(I)*XMU
           EPPP(I)=PVAS(I+9,IPOS_ICO)
6          CONTINUE
           ELSE IF (LIST(3,IPOS_ICO).EQ.2) THEN  !Satellite-Planet. ICO is a
               satellite
           DO 7 I=1,3
           E(I)=PVAS(I,IPOS_ICO)-PVAS(I,IPOS_IPLN)
           XT(I)=PVAS(I,IPOS_IPLN)+PVAS(I,IPOS_BARY)+E(I)*(XMU)
```

```
           EP(I)=PVAS(I+3,IPOS_ICO)-PVAS(I+3,IPOS_IPLN)
.................
C Computation of the scaling factor K and the matrix C=(C1,C2,C3)
        XKB=E(1)*E(1)+E(2)*E(2)+E(3)*E(3)
        XK=DSQRT(XKB)
        CALL VEC(E,EP,EA)
        XKD=EA(1)*EA(1)+EA(2)*EA(2)+EA(3)*EA(3)
        XKA=DSQRT(XKD)
...............
C Computation of the first derivative of K and C
        XKP=(E(1)*EP(1)+E(2)*EP(2)+E(3)*EP(3))/XK
        CALL VEC(E,EPP,EB)
        DO 22 I=1,3
22      C1P(I)=(-XK*EP(I)+XKP*E(I))/XKB
        RL=(C3(1)*EB(1)+C3(2)*EB(2)+C3(3)*EB(3))/XKA
...............
C Computation of the second derivative of K and C
        XKPP=EP(1)*EP(1)+EP(2)*EP(2)+EP(3)*EP(3)
        XKPP=XKPP+E(1)*EPP(1)+E(2)*EPP(2)+E(3)*EPP(3)
        XKPP=(XKPP-XKP*XKP)/XK
        XKE=(XKPP*XK-2.D0*XKP*XKP)/(XKB*XK)
        DO 50 I=1,3
50      C1PP(I)=-EPP(I)/XK+2.D0*EP(I)*XKP/XKB+E(I)*XKE
...............
C End of the computations

100     IF (IS.EQ.1) GOTO 200

C Transformation from adimensional to equatorial coordinates
        DO 19 I=1,3
19      EQ(I)=XK*(C1(I)*AD(1)+C2(I)*AD(2)+C3(I)*AD(3))+XT(I)
        IF (IPVA.EQ.1) RETURN
        DO 21 I=1,3
        R=C1(I)*AD(1)+C2(I)*AD(2)+C3(I)*AD(3)
        S=C1P(I)*AD(1)+C2P(I)*AD(2)+C3P(I)*AD(3)
        T=(C1(I)*AD(4)+C2(I)*AD(5)+C3(I)*AD(6))*ENEM
        EQ(I+3)=XT(I+3)+XKP*R+XK*(S+T)
        IF (IPVA.EQ.2) GOTO 21
        U=C1PP(I)*AD(1)+C2PP(I)*AD(2)+C3PP(I)*AD(3)
        UU=(C1P(I)*AD(4)+C2P(I)*AD(5)+C3P(I)*AD(6))*ENEM
        UUU=(C1(I)*AD(7)+C2(I)*AD(8)+C3(I)*AD(9))*ENEM*ENEM
        EQ(I+6)=XT(I+6)+XKPP*R+XK*(UUU+U)+2.D0*(XKP*(S+T)+XK*UU)
21      CONTINUE
        RETURN

C Transformation from equatorial to adimensional coordinates
200     DO 121 I=1,3
```

```
82          E(I)=EQ(I)-XT(I)
83          IF (IPVA.EQ.1) GOTO 121
84          EP(I)=EQ(I+3)-XT(I+3)
85          IF (IPVA.EQ.2) GOTO 121
86          EPP(I)=EQ(I+6)-XT(I+6)
87  121     CONTINUE
88          AD(1)=(C1(1)*E(1)+C1(2)*E(2)+C1(3)*E(3))/XK
89          AD(2)=(C2(1)*E(1)+C2(2)*E(2)+C2(3)*E(3))/XK
90          AD(3)=(C3(1)*E(1)+C3(2)*E(2)+C3(3)*E(3))/XK
91          IF (IPVA.EQ.1) RETURN
92          DO 122 I=1,3
93  122     AD(I+3)=-XKP*AD(I)
94          DO 123 I=1,3
95          AD(4)=AD(4)+C1(I)*EP(I)+C1P(I)*E(I)
96  ..............
```

2.6. Subroutine *dtranbl*. Here, we are going to explain subroutine "`dtranbl`" and similars.

In **"dtranbl"** we are going to transform from JPL-equatorial to addimensional coordinates (position and velocity) and computation of the differential of this transformation or viceversa. In this subroutine, we need a day and coordinates, for example JPL-equatorial coordinates (in km and km/day) with origin according to `IBC` (see common model), and it returns adimensional coordinates and the differential of this change. The same computations can be done for the inverse change.

Subroutine **"dtranbljplpl"** does the same computations but the transformation is from JPL-equatorial to JPL coordinates centered in the barycenter of a planet (`IBC`) (position and velocity) and computation of the differential of this transformation or viceversa.

Something similar happens in subroutine **"dtranblplrtbp"**. In this subroutine, we transform from JPL-equatorial coordinates centered in the barycenter of a planet (`IBC`) to addimensional coordinates (position and velocity) and computation of the differential of this transformation or viceversa.

```
1           SUBROUTINE DTRANBL(DAY,EQC,ADL,ICA,DTR,IS)
2   ..............
3   C
```

```
100     IF (IS.EQ.1) GOTO 200

C
C Transformation from adimensional to equatorial coordinates
C
        IF (ICA.EQ.1) THEN
        CALL TRANSLI(ADB,ADL,2,-1)
        ELSE
        DO 24 I=1,6
        ADB(I)=ADL(I)
24      CONTINUE
        ENDIF
        DO 19 I=1,3
        EQC(I)=XK*(C1(I)*ADB(1)+C2(I)*ADB(2)+C3(I)*ADB(3))+XT(I)
19      CONTINUE
        DO 21 I=1,3
        R=C1(I)*ADB(1)+C2(I)*ADB(2)+C3(I)*ADB(3)
        S=C1P(I)*ADB(1)+C2P(I)*ADB(2)+C3P(I)*ADB(3)
        T=(C1(I)*ADB(4)+C2(I)*ADB(5)+C3(I)*ADB(6))*ENEM
        EQC(I+3)=XT(I+3)+XKP*R+XK*(S+T)
21      CONTINUE
        DTR(1,1)=1.D0/ENEM
        DO 25 J=2,7
        DTR(1,J)=0.D0
25      CONTINUE
        DO 27 I=1,3
        DTR(I+1,2)=XK*C1(I)
        DTR(I+1,3)=XK*C2(I)
        DTR(I+1,4)=XK*C3(I)
.............
C
C Transformation from equatorial to adimensional coordinates
C
200     DO 121 I=1,3
        E(I)=EQC(I)-XT(I)
        EP(I)=EQC(I+3)-XT(I+3)
        EPP(I)=EQC(I+6)-XT(I+6)
121     CONTINUE
        ADB(1)=(C1(1)*E(1)+C1(2)*E(2)+C1(3)*E(3))/XK
        ADB(2)=(C2(1)*E(1)+C2(2)*E(2)+C2(3)*E(3))/XK
        ADB(3)=(C3(1)*E(1)+C3(2)*E(2)+C3(3)*E(3))/XK
        DO 122 I=1,3
        ADB(I+3)=-XKP*ADB(I)
122     CONTINUE
        DO 123 I=1,3
        ADB(4)=ADB(4)+C1(I)*EP(I)+C1P(I)*E(I)
        ADB(5)=ADB(5)+C2(I)*EP(I)+C2P(I)*E(I)
```

```
51          ADB(6)=ADB(6)+C3(I)*EP(I)+C3P(I)*E(I)
52  123     CONTINUE
53  ..................
54          DTR(5,I+4)=DTR(2,I+1)/ENEM
55          DTR(6,I+4)=DTR(3,I+1)/ENEM
56          DTR(7,I+4)=DTR(4,I+1)/ENEM
57          DTR(5,I+1)=(C1P(I)-XKP*C1(I)/XK)/XK
58          DTR(6,I+1)=(C2P(I)-XKP*C2(I)/XK)/XK
59          DTR(7,I+1)=(C3P(I)-XKP*C3(I)/XK)/XK
60  132     CONTINUE
61          DO 134 I=1,3
62          DTR(I+1,1)=DTR(I+4,2)*E(1)+DTR(I+4,3)*E(2)+DTR(I+4,4)*E(3)-
63      &           DTR(I+1,2)*XT(4)-DTR(I+1,3)*XT(5)-DTR(I+1,4)*XT(6)
64          DTR(I+4,2)=DTR(I+4,2)/ENEM
65          DTR(I+4,3)=DTR(I+4,3)/ENEM
66          DTR(I+4,4)=DTR(I+4,4)/ENEM
67          AM(1,I)=(C1PP(I)-2.D0*XKP*C1P(I)/XK)/XK
68          AM(2,I)=(C2PP(I)-2.D0*XKP*C2P(I)/XK)/XK
69          AM(3,I)=(C3PP(I)-2.D0*XKP*C3P(I)/XK)/XK
70  134     CONTINUE
71          U=(2.D0*XKP*XKP-XK*XKPP)/(XK*XK)
72          DO 136 I=1,3
73          EG(1)=U*DTR(I+4,5)+AM(I,1)/ENEM
74          EG(2)=U*DTR(I+4,6)+AM(I,2)/ENEM
75          EG(3)=U*DTR(I+4,7)+AM(I,3)/ENEM
76  ..................
```

3. Short file descriptions

Let's do a brief summary of the subroutines that we have in the package. To do this, we grouped the subroutines in blocks to indicate which of them do similar work:

- Read ephemerides: We use these programs to read ephemerides files (*.bsp) or to choose the bodies we want to select.
  - **model.dat**: This is a text file where we provide the model (epoch, origin of coordinates, reference system and equilibrium point) that we are going to use.
  - **modeleph.dat**: Another text file, here we have all the bodies and we can select which we want to include (with a "1").
  - **nmodjpl**:This routine is used for the initialization of all the basic constants such as the mass ratios of the bodies, semiaxes, and the JPL-model of the solar system adopted.

  - **read_modeleph**: It reads the bodies that we select in `modeleph.dat` and we will write these in `new_files.dat`.
  - **change_file_dat**: It creates a new file, `new_files.dat` with the selected bodies.
  - **commonmodel.inc**: This file contains common variables that are shared between several subroutines.
- Manipulate list: In this block we are going to describe the subroutines that use the list of some way.
  - **barorcom**: It changes the mode (activate or desactivate) of a selected body in the list.
  - **cong**: It looks if the list is consistent with the planets and satellites selected.
  - **dessat**: To activate or deactivate a satellite.
  - **order**: It checks if the elements of list are ordered. If they aren't, it orders the list by binary insertion.
  - **putlist**: It puts elements ordered in list by binary insertion.
  - **rmvlist**: To remove an element of the list.
  - **writelist**: It writes the bodies selected in file *"list.dat"*.
- Computations: Block to do differents calculations that we are going to use.
  - **cdjta**: It changes Julian Dates into adimensional RTBP time or viceversa.
  - **factgamma**: It calculates the scaling factor gamma.
  - **fillsemiaxes**: This subroutine fills the semiaxes of the orbits of the bodies of the solar system in km. Values taken from Explanatory Supplement ([**9**]).
  - **fillxmue**: It fills a variable with the values *GM* of all bodies. In km^3/s^2.
  - **plajplnsat**: Routine for the computation of the position, velocity, acceleration and overacceleration of the bodies of the solar system using the JPL ephemerides.
  - **xmu_enem**: This routine calculate the mass parameter of the RTBP considered and the mean motion of the system.
  - **spkg**: Auxiliary routine used in `plajplnsat` to calculate state of a body.
  - **spke0203ipv**: Auxiliary routine used in `spkg` to calculate acceleration and overacceleration.
  - **vec**: Auxiliary routine to compute the cross product of 3 by 3 vectors.
- Vector fields: These routines implement some vector fields that we will explain later.
  - **derivcs**: It computes the vector field defining the equations of motion and the variational equations with respect to time and coordinates (if wanted) for a

particle in the the solar system according to a given model centered in some place of the solar system.

- **dervin**: Another subroutine to compute another vector field in inertial coordinates.
- **dervsin**: Another subroutine to compute another vector field in synodic coordinates.
- **rk78**: This routine is an implementation of a Runge-Kutta-Fehlberg method of orders 7 and 8.

• Change of coordinates: This block contains subroutines to accomplish different changes of coordinates.

- **transs / transsk1**: Transformation from adimensional RTBP coordinates to JPL coordinates or viceversa. In `transsk1` the scaling distance factor is $k = 1$.
- **transsinsin**: Transformation from inertial to sinodical coordinates or viceversa.
- **transsjplpl / transsjplplk1**: Transformation from JPL coordinates centered in Solar System Barycentre to JPL coordinates centered in the barycenter of a planet or viceversa.
- **transsplrtbp / transsplrtbpk1**: Transformation from adimensional RTBP coordinates to JPL coordinates centered in the barycenter of a planet or viceversa.
- **dtranbl**: Transformation from JPL-equatorial to addimensional coordinates (position and velocity) and computation of the differential of this transformation or viceversa.
- **dtranbljplpl**: Transformation from JPL-equatorial to JPL coordinates centered in the barycenter of a planet (position and velocity) and computation of the differential of this transformation or viceversa.
- **dtranblplrtbp**: Transformation from JPL-equatorial coordinates centered in the barycenter of a planet to addimensional coordinates (position and velocity) and computation of the differential of this transformation or viceversa.
- **transli**: Transformation from adimensional barycentric coordinates in a RTBP frame to addimensional-normalized coordinates with origin at an equilibrium point or viceversa.
- **trcabj**: This is an auxiliary routine to change from any type of coordinates to another.

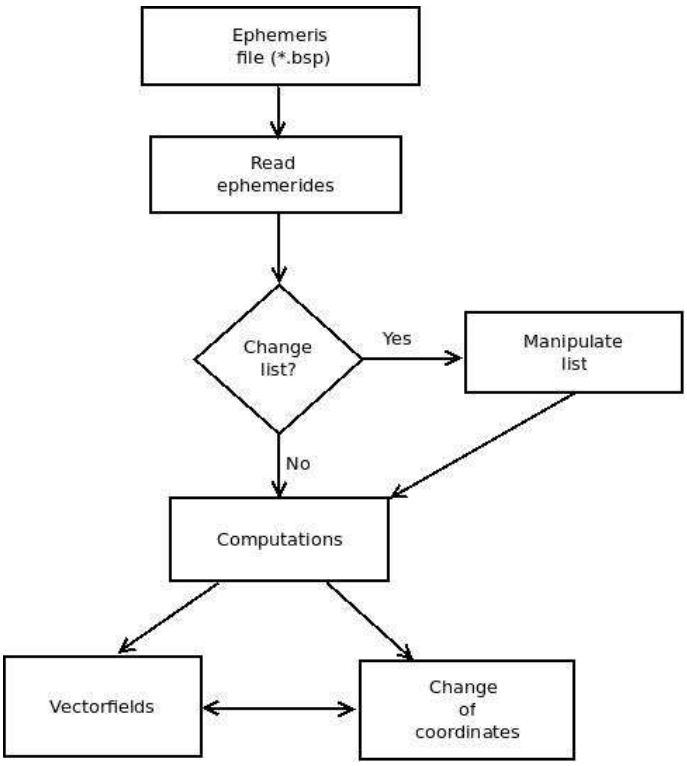

FIG. 6. Workflow diagram of blocks.

Chapter 4
Use of the software. Simulations and Results

The objetive of this chapter is to show how to use this software and to test the software package, doing several simulations of the differents programs. First, we are going to check if routines of change of coodinates and of vector fields are correct. After this, we will do some integrations of orbits to see the figures that appear. Finally, we will show one of the applications of the program, the parallel shooting.

1. How to use this software package

We need to select a determinate model as predetermined. This model is stored at the file `list.dat`. If we would want change this model, first we should have to open `modeleph.dat` with a simple text editor, and here, put in the second column a "1" value on the bodies that we want choose and change into a "0" value the bodies that we don't want to choose for the model. Also, we have to open the file `model.dat` and select the conditions of the model with the parameters that we have explained for this file before.

With this, we can run the main program, executing next command on the prompt:

```
$ ./main.exe
```

This command will run the main routine selected, that will call always to the subroutine `nmodjpl`. We should remember, as we have explained, that in this subroutine there are several calls to routines commented. If we would want use some of these routines, we only have to uncomment them in `nmodjpl`, and compile again the program with the command:

```
$ ./make.csh
```

Finally, we have built different "main programs" to use all the subroutines of vector fields and change of coordinates that we have. But, if we would want to use subroutines otherwise we would only have to follow the examples of these main programs (all of them are named like "`main_*.f`".

2. Tests

The tests that we are going to do check the change of coordinates in different systems in order to see whether they are correct.

2.1. Changing coordinates from RTBP to JPL. Our first test consists in checking the subroutine **transs**. We should remember that this subroutine makes a change of coordinates from RTBP to JPL coordinates or viceversa. Then, this test will consist in take coordinates in a JPL system and do the change to RTBP, and after, this RTBP coordinates change them again to JPL coordinates, and check that they were the same than at first.

We take initial conditions:

$X = -0.2656698311 \cdot 10^8, Y = \ \ 0.1340162335 \cdot 10^9, Z = \ \ 0.5828199141 \cdot 10^8$

$\dot{X} = -0.2583199517 \cdot 10^7, \dot{Y} = -0.4156446631 \cdot 10^6, \dot{Z} = -0.1802176846 \cdot 10^6$

$\ddot{X} = -0.2656698311 \cdot 10^5, \ddot{Y} = \ \ 0.1340162335 \cdot 10^6, \ddot{Z} = \ \ 0.5828199141 \cdot 10^5$

Basically, the main Fortran code is below:

```
      INCLUDE 'commonmodel.inc'
      CALL NMODJPL

      N=9
      DAY=0.D0
      IPVA=3
      OPEN (NCD,FILE=ARXOUT)

      IS=1
      CALL TRANSS(DAY,EQ,AD,IPVA,IS)

      IS=-1
      CALL TRANSS(DAY,EQB,AD,IPVA,IS)

      WRITE(NCD,*) 'DIFFERENCE: '
      DO I=1,N
       DIF(I)=EQB(I)-EQ(I)
```

```
18          END DO
19          WRITE(NCD,100)  (DIF(I),I=1,N)
```

What we are doing here is to call to the subroutine **transs** with value `IS=1` and `IPVA=3` for that in this way it makes a change of positions, velocities and accelerations from JPL coordinates to RTBP ones at time `0` in Julian ephemerides epoch. After, with `IS=-1` we make the inverse change.

When we call to **nmodjpl** we read the JPL model of the Solar System that we are adopting. In this first test, our model is `ICO=3` (Earth Barycentre), and bodies: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, Moon and Sun. The results are:

```
DIFFERENCE:
 0.3725290298E-08 -0.2980232239E-07  0.0000000000E+00
 0.4656612873E-09 -0.1164153218E-09 -0.5820766091E-10
 0.7275957614E-11 -0.2910383046E-10  0.7275957614E-11
```

If we change the model, `ICO=599` (i.e. Jupiter) and bodies: Jupiter, Sun, Moon, Io and Europa; we obtain:

```
DIFFERENCE:
-0.4470348358E-07 -0.1490116119E-07 -0.3725290298E-07
 0.0000000000E+00 -0.1746229827E-09 -0.2910383046E-09
 0.1091393642E-10 -0.2910383046E-10 -0.1455191523E-10
```

As we can observe, the difference is very small.

2.2. Routines of vector fields and transformation of coordinates from inertial to sinodical coordinates. In this test we will check the subroutines **transsinsin**, **dervin** and **dervsin**. We will test the diagram:

$$\begin{array}{ccc} (R,\dot{R}) & \longrightarrow & (R,\dot{R},\ddot{R}) \\ \downarrow transsinsin & & \uparrow \\ (r,\dot{r}) & \longrightarrow & (r,\dot{r},\ddot{r}) \end{array}$$

We take as initial conditions:
$X = -0.2656698311 \cdot 10^8, Y = \ \ 0.1340162335 \cdot 10^7, Z = \ \ 0.5828199141 \cdot 10^7$

$\dot{X} = -0.2583199517 \cdot 10^7$, $\dot{Y} = -0.4156446631 \cdot 10^6$, $\dot{Z} = -0.1802176846 \cdot 10^6$
and time:
DAY=0.0, at Julian ephemerides epoch.

The main Fortran code is:

```
      INCLUDE 'commonmodel.inc'
      CALL NMODJPL

      N=6

      OPEN (NCD,FILE=ARXOUT)
      CALL DERVIN(TI,XI,N,F)
      XI(7)=F(4)
      XI(8)=F(5)
      XI(9)=F(6)
      CALL TRANSSINSIN(TI,XI,AD,3,1)
      CALL DERVSIN(TI,AD,N,F(4))
      F(1)=AD(1)
      F(2)=AD(2)
      F(3)=AD(3)
      DO I=1,9
       DIF(I)=F(I)-AD(I)
      END DO
```

What we are doing in this program is, first we call to subroutine **dervin** to calculate the acceleration of the system ($\ddot{R}$) as we have discussed in section 2.2. After this, we call to **transsinsin** to transformate from inertials to sinodical coordinates, the position, velocity and acceleration, so that, when we call to **dervsin** to calculate the acceleration ($\ddot{r}$) as 2.2, we can check if the diagram closes.

If we take as model ICO=399 (Earth), and bodies: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, Moon and Sun. The results are:

```
 (R1,R1P):
 -0.2656698311E+08  0.1340162335E+07  0.5828199141E+07
 -0.2583199517E+07 -0.4156446631E+06 -0.1802176846E+06
(R1,R1P,R1PP):
 -0.2656698311E+08  0.1340162335E+07  0.5828199141E+07
 -0.2583199517E+07 -0.4156446631E+06 -0.1802176846E+06
  0.5513041339E+04  0.7341635950E+04  0.1824414001E+04
r, trasformed from R to r:
```

```
  0.8048470006E+07  0.2556588525E+08  0.4814163878E+07
  0.4512234215E+06  0.2479470000E+07 -0.2205203149E+03
  0.9722118781E+05 -0.1468843896E+05 -0.1281048173E+04
(r2,r2p,r2pp):
  0.8048470006E+07  0.2556588525E+08  0.4814163878E+07
  0.4512234215E+06  0.2479470000E+07 -0.2205203149E+03
  0.9722118781E+05 -0.1468843896E+05 -0.1281048173E+04
DIFFERENCE:
  0.0000000000E+00  0.0000000000E+00  0.0000000000E+00
  0.0000000000E+00  0.0000000000E+00  0.0000000000E+00
  0.0000000000E+00 -0.1818989404E-11  0.1591615728E-11
```

When we change the model, with `ICO=699` (Saturn), several of their moons, the Sun and the Moon, we obtain:

```
 (R1,R1P):
 -0.2656698311E+08  0.1340162335E+07  0.5828199141E+07
 -0.2583199517E+07 -0.4156446631E+06 -0.1802176846E+06
(R1,R1P,R1PP):
 -0.2656698311E+08  0.1340162335E+07  0.5828199141E+07
 -0.2583199517E+07 -0.4156446631E+06 -0.1802176846E+06
  0.4054344113E+03 -0.6262105298E+02 -0.8972787062E+02
r, trasformed from R to r:
 -0.1619457957E+08  0.2155771772E+08  0.3816926892E+07
 -0.2112956907E+07  0.1541089824E+07 -0.1101373260E+06
  0.2156594447E+04  0.2383921360E+04 -0.3566036065E+03
(r2,r2p,r2pp):
 -0.1619457957E+08  0.2155771772E+08  0.3816926892E+07
 -0.2112956907E+07  0.1541089824E+07 -0.1101373260E+06
  0.2156594447E+04  0.2383921360E+04 -0.3566036065E+03
DIFFERENCE:
  0.0000000000E+00  0.0000000000E+00  0.0000000000E+00
  0.0000000000E+00  0.0000000000E+00  0.0000000000E+00
  0.1158696250E-08  0.4900357453E-08  0.3763034329E-09
```

We can appreciate, in both examples, that the difference is very small.

Now, we are going to do the same, but instead of to calculate the acceleration only with the field, we are going to integrate (using the fields of section 2.2) for a given time span. We take the same initial conditions, and for integrations, we take the conditions:
`HMI`=$1.0 \cdot 10^{-1}$, the minimum allowed value for the absolute value of `H`.
`HMA`=$1.0 \cdot 10^{-1}$, the maximum allowed value for the absolute value of `H`.

H=$1.0 \cdot 10^{-1}$, the time step to be used.
TI=0.0, initial time in Julian ephemerides epoch.
TF=14.331103, final time in Julian ephemerides epoch.
$E_1 = 1. \cdot 10^{-13}$, tolerance.

```
      INCLUDE 'commonmodel.inc'
      EXTERNAL DERVIN
      EXTERNAL DERVSIN
      CALL NMODJPL

      CALL TRANSSINSIN (TI,XI,AD1,2,1)
 5      CALL RK78(TI,XI,N,H,HMI,HMA,E1,R,B,F,DERVIN)
      IF (TI.LT.TF) GOTO 5
      CALL TRANSSINSIN(TI,XI,AD2,2,1)
 6      CALL RK78(TI,AD1,N,H,HMI,HMA,E1,R,B,F,DERVSIN)
      IF (TI.LT.TF) GOTO 6

      DO I=1,9
       DIF(I)=AD2(I)-AD1(I)
      END DO
```

If we take as model ICO=399 (Earth), and bodies: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, Moon and Sun. The results are:

```
DIFFERENCE:
  0.3296881914E-05  0.2731382847E-04 -0.2572955564E-03
  0.9997747838E-06  0.4198402166E-05 -0.4010281737E-04
```

For the states that we have taken (of order of 10^8), the difference can be considered small.

3. Integrations

We are going to integrate the vector field of **"derivcs"** to see the orbits that we can obtain, with initial conditions:

$$X = -1.01126785748736, Y = 0.0, \qquad Z = 9.069633331713 \cdot 10^{-04}$$
$$\dot{X} = -0.0, \qquad \dot{Y} = 9.073734203283 \cdot 10^{-03}, \dot{Z} = 0.0.$$

The main code that we will to use is:

```
      INCLUDE 'commonmodel.inc'
```

```
      EXTERNAL DERIVCS
      CALL NMODJPL
C     Initial condition
      TRI=0.D0
      PERI=2.5D0
c Initial condition and time from RTBP to JPL
      CALL CDJTA(TJI,TRI,ORIG,-1)
      TRF=TRI+PERI
      CALL CDJTA(TJF,TRF,ORIG,-1)
      IPVA=2  ! pos+vel
      CALL TRANSS(TJI,XJI,XRI,IPVA,-1)
50    CALL RK78 (DIA,XJP,N,H,HMIN,HMAX,TOLRK,R,B,F,DERIVCS)
      CALL CDJTA(DIA,TRP,ORIG,1)
      CALL TRANSS(DIA,XJP,XRT,IPVA,1)
      IF (DIA.LT.TJF) GOTO 50
```

If we take as model `ICO=12` (Earth+Moon Barycentre), and bodies: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, Moon and Sun. The results are:

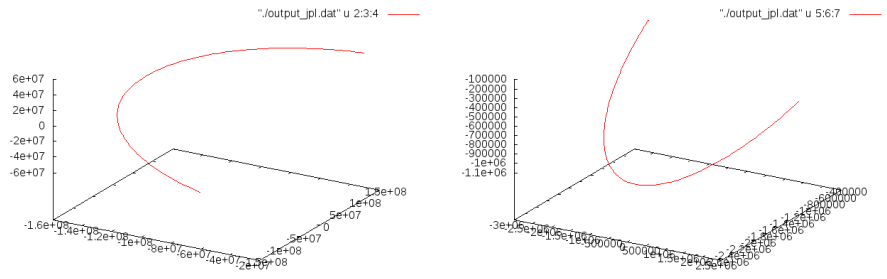

FIG. 1. Orbit in coordinates JPL.

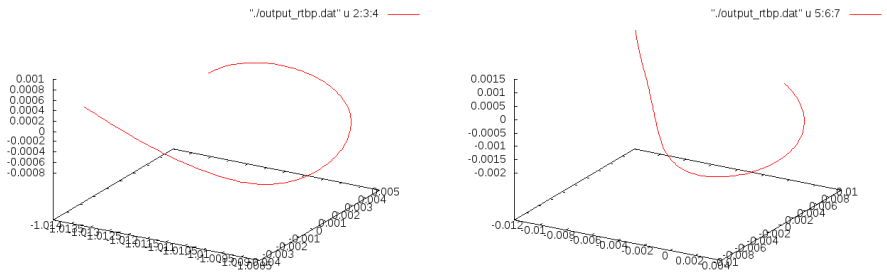

FIG. 2. Orbit in coordinates RTBP.

4. Application to calculate of a Quasi-Periodic orbit in the Solar System

One of the applications of this software, once we have tested it for a simple model, is the parallel shooting, that consists in doing several refinements of a integration, from a given initial seed.

4.1. Parallel shooting. In this section we are going to get solutions of more realistic equations of motion (i.e., Newton's equations using some JPL model for the motion of the bodies of the solar system). As a first restriction, considering that we are going to compute these solutions numerically, we will fix a time span, which means we will fix an initial epoch (now the system is non autonomous) and a length for the time span.

The general idea is to use a multiple shooting method similar to the one used for the numerical solution of boundary-value problems. This method is useful for the computation of highly unstable periodic orbits with very long periods. Our problem is, in some sense, close to this one. As in the standard procedure, first we split the total time span into a number of shorter subintervals selecting, for instance, `N` equally spaced points $\mathtt{t}_1, \mathtt{t}_2, \ldots, \mathtt{t}_N$ ($\mathtt{t}_1$ is the initial epoch and $\mathtt{t}_N - \mathtt{t}_1$ the length of the time interval mentioned above). Different time intervals can be used, but we have chosen here all of them equal. Denote by $\Delta t = t_{i+1} - t_i$ and by

$$Q_i = (t_i, x_i, y_i, z_i, \dot{x}_i, \dot{y}_i, \dot{z}_i)^T, i = 1, 2, ...N$$

the points on a fixed quasihalo orbit of the RTBP, equally spaced (Δt) in time, computed using the formal expansions. Let $\phi(Q_i)$ be the image of the point $\mathtt{Q}_i$ under the flow associated to the equations of motion in the solar system after an amount of time Δt. As, in this way, the epochs $\mathtt{t}_i$ are fixed, we can write $Q_i = (x_i, y_i, z_i, \dot{x}_i, \dot{y}_i, \dot{z}_i)^T$. If all the points $\mathtt{Q}_i$ would be on the same orbit of the new equations, then $\phi(Q_i) = Q_{i+1}$ for $\mathtt{i=1,\ldots,N-1}$.

As this is not the case, we need to change the starting values. In this way, we must solve a set of $\mathtt{N-1}$ nonlinear equations, which can be written as

$$F \begin{pmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_N \end{pmatrix} = \begin{pmatrix} \phi(Q_1) \\ \phi(Q_2) \\ \vdots \\ \phi(Q_{N-1}) \end{pmatrix} - \begin{pmatrix} Q_2 \\ Q_3 \\ \vdots \\ Q_N \end{pmatrix} = \phi \begin{pmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_{N-1} \end{pmatrix} - \begin{pmatrix} Q_2 \\ Q_3 \\ \vdots \\ Q_N \end{pmatrix} = 0.$$

We use Newton's method to solve the above system. If $Q^{(j)} = (Q_1^{(j)}, Q_2^{(j)}, ..., Q_N^{(j)})^T$, denotes hte `j`-th iterate of the procedure, Newton's equations can be written as

$$DF(Q^{(j)}) \cdot (Q^{(j+1)} - Q^{(j)}) = -F(Q^{(j)}),$$

where the differential of the funcion `F` has the following structure

$$DF = \begin{pmatrix} A_1 & -I & & & \\ & A_2 & -I & & \\ & & \ddots & \ddots & \\ & & & A_{N-1} & -I \end{pmatrix},$$

and

$$D\phi = \begin{pmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_{N-1} \end{pmatrix}.$$

As each of the transition matrices, $\mathtt{A}_i$, that appear in $D\phi$ are 6×6, at each step of the method we have to solve a system of $(N-1) \times 6$ equations with $6 \times N$ unknowns, so some additional conditions must be added. This is the only difference with the standard multiple shooting method and is due to the fact that our problem is not a real boundary-value one. As additional equations we could fix some initial and final conditions at `t=`$\mathtt{t}_0$ and `t=`$\mathtt{t}_N$. In this case one must take care with the choice because the problem can be ill conditioned from the numerical point of view. This is because the matrix `DF(Q)` has a very large condition number. If we assume that Δt is of the order of one revolution, all the matrices $\mathtt{A}_i$ are similar and if the largest eigenvalue is λ, the smallest is λ^{-1} , so the condition number is of the order of λ^2. To avoid this bad conditioning, we can choose a small value for Δt. In this case the largest eigenvalue of $\mathtt{A}_i$ is not so large, but as the number of points $\mathtt{Q}_i$ increases (if we want to cover the same time span) the instability is transfered to the procedure for solving the linear system. Also, the extra boundary conditions can force the solution in a non natural way giving convergence problems when we try to compute the orbit for a long time interval.

To avoid this, we can apply Newton's method directly. As the system has more unknowns than equations, we have (in general) an hyperplane of solutions. From this set

of solutions we try to select the one closer to the initial orbit used to start the procedure. This is done by requiring the correction to be minimum with respect to some norm (i.e. the euclidean norm). The use of the normal equations must be avoided because they are usually ill conditioned too. In our situation we have used the structure of the equations in the following way.

Denoting by $\Delta Q^{(j)}$

$$\Delta Q^{(j)} = \Delta Q^{(j+1)} - \Delta Q^{(j)},$$

if we require $\|\Delta Q^{(j)}\|_2$ to be minimum, using the Lagrange function $L(\Delta Q, \mu)$ with (vector) multiplier μ

$$L(\Delta Q, \mu) = \Delta Q^T \cdot \Delta Q + \mu^T \cdot (F(Q) + DF(Q) \cdot \Delta Q),$$

we get

$$\Delta Q^{(j)} = -DF(Q^{(j)})^T \cdot [DF(Q^{(j)}) \cdot DF(Q^{(j)})^T]^{-1} \cdot F(Q^{(j)}), \tag{11}$$

which gives the value of $\Delta Q^{(j)}$ explicitly. Let $M = DF(Q^{(j)}) \cdot DF(Q^{(j)})^T$. This is a symmetric band matrix with the following pattern

$$M = \begin{pmatrix} I + A_1 A_1^T & -A_1 & & & & \\ -A_1^T & I + A_2 A_2^T & -A_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & -A_{N-3}^T & I + A_{N-2} A_{N-2}^T & -A_{N-2} & \\ & & & -A_{N-2}^T & I + A_{N-1} A_{N-1}^T & \end{pmatrix}.$$

We introduce additional variables, $Z^{(j)}$, by $M^{-1} F(Q^{(j)}) = Z^{(j)}$, and then equation 11 becomes

$$M \cdot Z^{(j)} = F(Q^{(j)}), \tag{12}$$

$$\Delta Q^{(j)} = -DF(Q^{(j)})^T \cdot Z^{(j)}. \tag{13}$$

Now we use Cholesky factorization to express `M` as

$$\begin{pmatrix} I & & & \\ L_2 & I & & \\ & \ddots & \ddots & \\ & & L_{N-1} & I \end{pmatrix} \cdot \begin{pmatrix} D_1 & & & \\ & D_2 & & \\ & & \ddots & \\ & & & D_{N-1} \end{pmatrix} \cdot \begin{pmatrix} I & L_2^T & & \\ & \ddots & \ddots & \\ & & I & L_{N-1}^T \\ & & & I \end{pmatrix},$$

obtaining the following recursive relations

$$\begin{aligned} D_1 &= I + A_1 A_1^T, \\ L_i &= -A_i D_{i-1}^{-1}, \qquad\qquad i = 2, 3, ..., N-1 \\ D_i &= I + A_i A_i^T - L_i D_{i-1} L_i^T, i = 2, 3, ..., N-1. \end{aligned}$$

With this factorization, system 12 can be solved recursively using intermediate variables, $\mathtt{X}$, $\mathtt{Y}$, with block components $\mathtt{X}_1, \ldots, \mathtt{X}_{N-1}, \mathtt{Y}_1, \ldots, \mathtt{Y}_{N-1}$ such that $\mathtt{Y}=\mathtt{Z}^{(j)}$ and $\mathtt{F}=(\mathtt{F}_1, \ldots, \mathtt{F}_{N-1})$

$$\begin{aligned} X_1 &= F_1(Q^{(j)}), \\ X_k &= F_k(Q^{(j)}) - L_k X_{k-1}, k = 2, 3, ..., N-1 \\ Y_{N-1} &= D_{N-1}^{-1} X_{N-1}, \\ Y_k &= D_k^{-1} X_k - L_{k+1}^T Y_{k+1}, k = N-2, N-3, ..., 1. \end{aligned}$$

The value of $\Delta Q^{(j)}$ can be computed using 13. From these last equations you must be careful with matrices $\mathtt{D}_k$ that appear in the recursive computation of $\mathtt{X}_k$ and $\mathtt{Y}_k$.

4.1.1. *Simulation.* To do simulations of parallel shooting it need a file with a 'initial seed' for each case indicating which is the amplitude of the orbit. This file has the form:

```
% INITIAL NODES FOR PARALL. SHOOT. HALO ORBIT FROM LP
 0.00000000000000000E+00  0.3040423398444176E-05 12  2  1  1   500
 1  0.0000000000E+00  0.1178598970650897E+00  0.0000000000000000E+00
     -0.1126259568615726E+00  0.0000000000000000E+00 -0.9071113988967449E+00
     0.0000000000000000E+00
 2  0.7000000000E+01  0.1118270637747303E+00 -0.1082006756254275E+00
     -0.1101348393324709E+00 -0.9939905240374050E-01 -0.8815349516041047E+00
     0.4123552402788498E-01
 3  0.1400000000E+02  0.9430067961777955E-01 -0.2103088925768840E+00
     -0.1027621787688251E+00 -0.1893949587537312E+00 -0.8064677054965651E+00
     0.8080594356953390E-01
 4  0.2100000000E+02  0.6692699208278957E-01 -0.3006212405867519E+00
     -0.9080503762373199E-01 -0.2617269755118253E+00 -0.6866198393466235E+00
     0.1171257425174262E+00
 5  0.2800000000E+02  0.3223093094510870E-01 -0.3741501783347533E+00
     -0.7474433499936742E-01 -0.3101854836989055E+00 -0.5290253048385434E+00
     0.1487284026901428E+00.............
```

The second line contains, in order: initial time, value of μ, `ICO` taked, value of the equilibrium point, type of coordinates (1 1) and number of nodes.

For the parallel shooting, we take periodic orbits, and we repeat the orbit several times. So that, the rest of lines of the file contain the node `i`, time, position and velocity for this time.

Below, we will show results for different simulations taking different initial conditions.

If we make a simulation of the parallel shooting, with the conditions `ICO=12` (Earth+Moon Barycentre), and bodies: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, Moon and Sun, we obtain:

- In the first iteration:

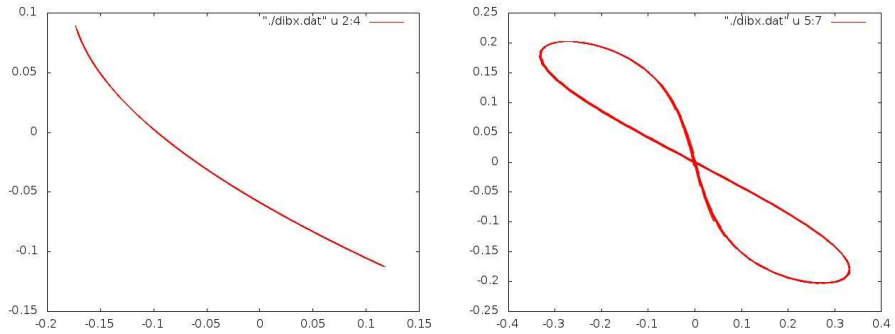

FIG. 3. Parallel shooting in Earth+Moon Barycentre.

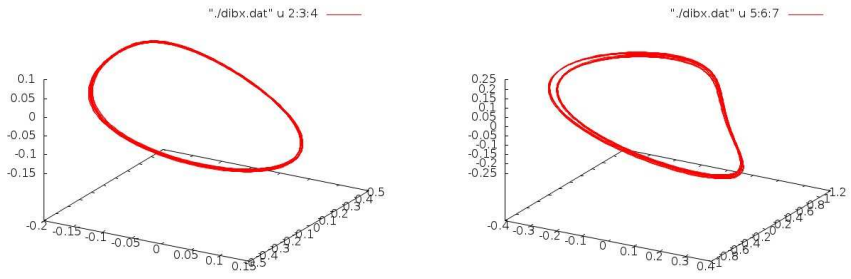

FIG. 4. Parallel shooting in Earth+Moon Barycentre.

```
NORM OF THE FUNCTION:     11898.8184
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1144595E+05     0.3251484E+04
     0.0000000E+00     0.9599395E+03     0.2654420E+03
     0.0000000E+00     0.9184081E+02     0.2494252E+02
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   477129.089
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4768051E+06     0.1757930E+05
     0.0000000E+00     0.4929047E+05     0.1388243E+04
     0.0000000E+00     0.1675518E+04     0.1604044E+03
```

- In the second iteration:

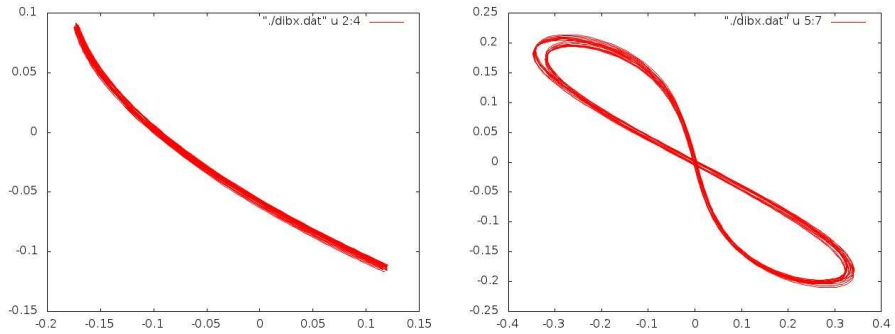

FIG. 5. Parallel shooting in Earth+Moon Barycentre.

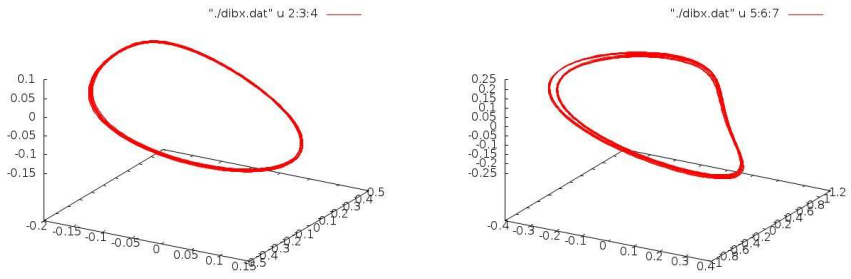

FIG. 6. Parallel shooting in Earth+Moon Barycentre.

```
NORM OF THE FUNCTION:     547.086757
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.5258641E+03     0.1509003E+03
     0.0000000E+00     0.1049855E+03     0.3038294E+02
     0.0000000E+00     0.4975914E-01     0.1171015E-01
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   30077.538
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3006367E+05     0.9132934E+03
     0.0000000E+00     0.3538269E+04     0.1016101E+03
     0.0000000E+00     0.7812096E+02     0.2364083E+01
```

- In the third iteration:

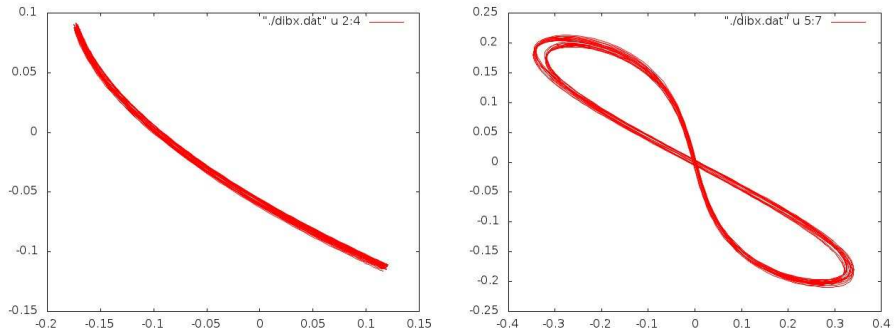

FIG. 7. Parallel shooting in Earth+Moon Barycentre.

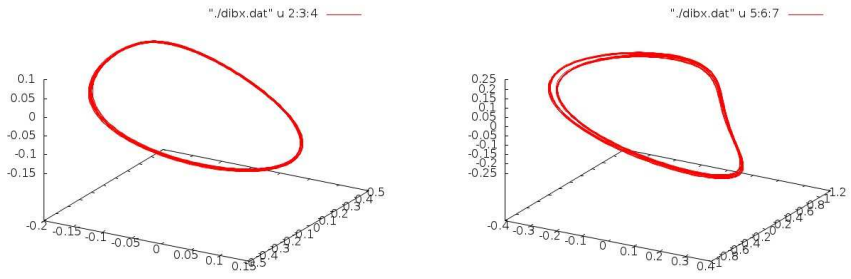

FIG. 8. Parallel shooting in Earth+Moon Barycentre.

```
NORM OF THE FUNCTION:     1.35948577
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1306491E+01     0.3758751E+00
     0.0000000E+00     0.2829894E+00     0.8152813E-01
     0.0000000E+00     0.1837883E-03     0.5229199E-04
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   105.221174
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1051741E+03     0.3146999E+01
     0.0000000E+00     0.1310183E+02     0.3633340E+00
     0.0000000E+00     0.7146532E+00     0.8468568E-02
```

- In the fourth iteration:

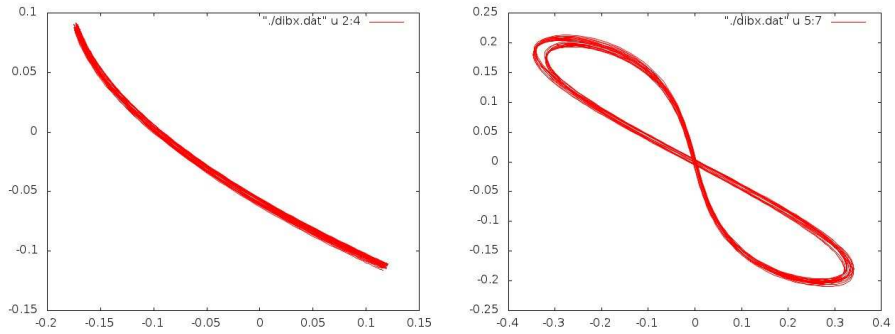

FIG. 9. Parallel shooting in Earth+Moon Barycentre.

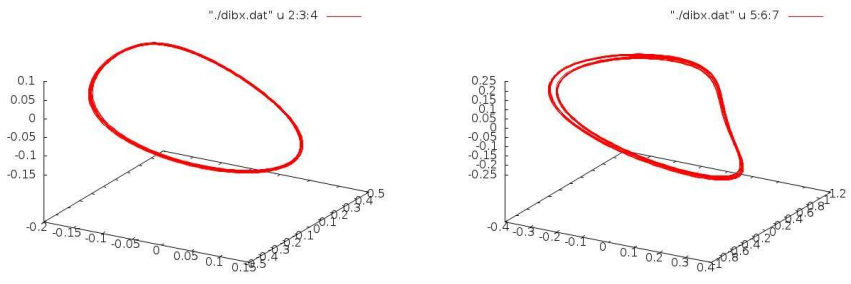

FIG. 10. Parallel shooting in Earth+Moon Barycentre.

```
NORM OF THE FUNCTION:     2.17353133E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2090841E-04     0.5938204E-05
     0.0000000E+00     0.7440528E-05     0.2156530E-05
     0.0000000E+00     0.2787752E-07     0.2818718E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   0.00111980696
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1119306E-02     0.3347524E-04
     0.0000000E+00     0.1608678E-03     0.4617990E-05
     0.0000000E+00     0.2760375E-05     0.7762607E-07
```

In each iteration, the norm of correction is reduced, and we have a result more precise.

Now, we are going to show other simulations with other initial conditions. These simulations we will display halo orbits. A halo orbit is a three-dimensional periodic orbit near the L_1, L_2 or L_3 Lagrange point. In our case, we use the Lagrange point L_1 and L_2. A spacecraft in a halo orbit does not technically orbit the Lagrange point itself (which is just an equilibrium point with no mass), but travels in a closed, repeating path near the Lagrange point.

- We take Jupiter with his natural satellite Io (`ICO=501`), with value of $\mu = 0.4704237643000000 \cdot 10^{-04}$. First, we take the Lagrange point L_1, and the value of γ for L_1 is $\gamma = 0.2481942967450895 \cdot 10^{-01}$ (and amplitude 0.1). With these conditions we create the initial seed, and we do the parallel shooting with results:

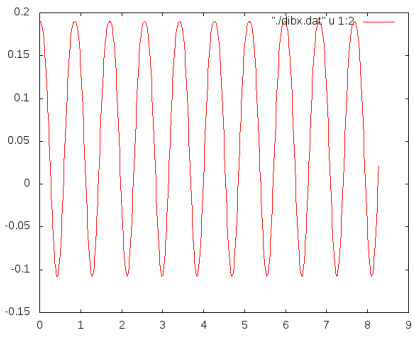

FIG. 11. Time used at parallel shooting in L_1 for Io-Jupiter.

– First iteration:

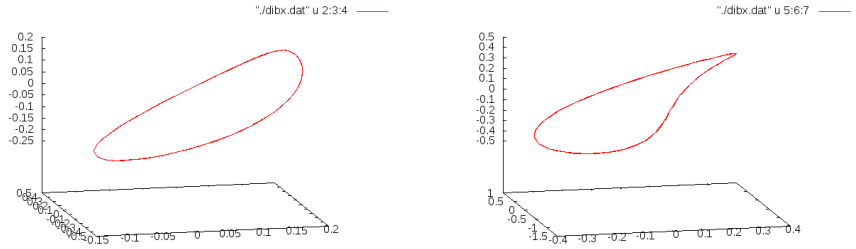

FIG. 12. Parallel shooting in L_1 for Io-Jupiter.

```
NORM OF THE FUNCTION:     1866.12102
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2641520E+02     0.1865934E+04
     0.0000000E+00     0.1958758E+01     0.1388491E+03
     0.0000000E+00     0.0000000E+00     0.1692552E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   15117.9377
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2243440E+04     0.1495055E+05
     0.0000000E+00     0.2411364E+03     0.1573064E+04
     0.0000000E+00     0.2060475E+02     0.4252557E+02
```

– Second iteration:

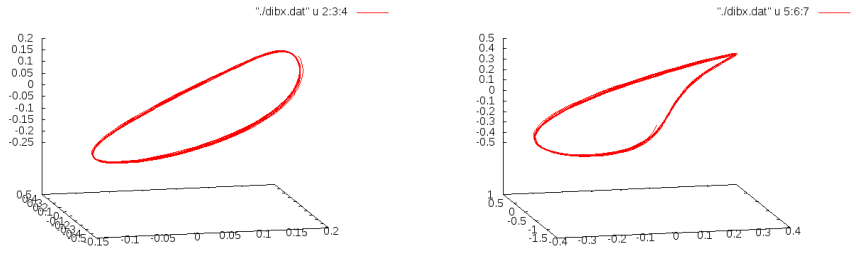

FIG. 13. Parallel shooting in L_1 for Io-Jupiter.

```
NORM OF THE FUNCTION:     70.2384356
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.9956970E+00     0.7023138E+02
     0.0000000E+00     0.2693221E+00     0.1922797E+02
     0.0000000E+00     0.1228781E-06     0.5277841E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   370.64696
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.5725213E+02     0.3661985E+03
     0.0000000E+00     0.6348275E+01     0.4372009E+02
     0.0000000E+00     0.1953608E+00     0.1411184E+01
```

– Last iteration:

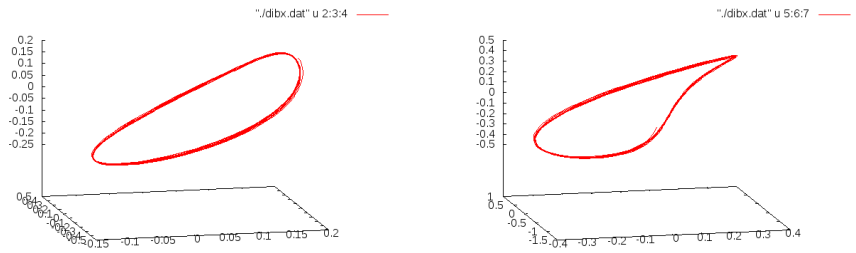

FIG. 14. Parallel shooting in L_1 for Io-Jupiter.

```
NORM OF THE FUNCTION:     5.99282498E-06
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4329550E-05     0.4143543E-05
     0.0000000E+00     0.7183534E-06     0.1096791E-05
     0.0000000E+00     0.0000000E+00     0.1809200E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   0.000165736878
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2568631E-04     0.1637343E-03
     0.0000000E+00     0.3369432E-05     0.2132566E-04
     0.0000000E+00     0.2656366E-06     0.1373295E-05
```

Now, we take the Lagrange point L_2 with $\gamma = 0.2523705103220314 \cdot 10^{-01}$ (and amplitude 0.08).

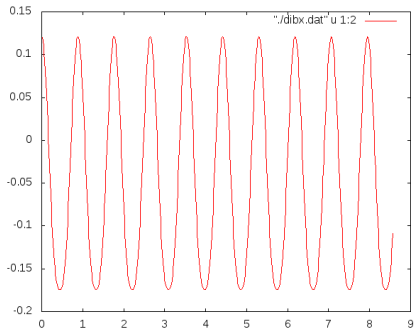

FIG. 15. Time used at parallel shooting in L_2 for Io-Jupiter.

– First iteration:

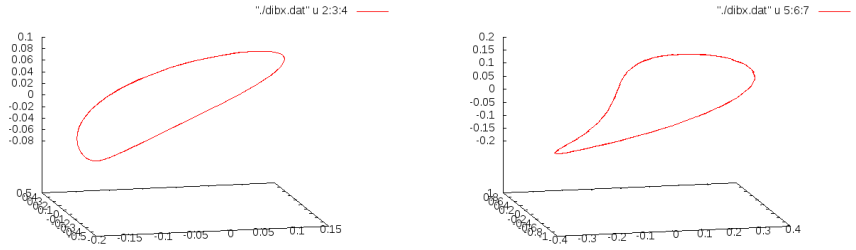

FIG. 16. Parallel shooting in L_2 for Io-Jupiter.

```
NORM OF THE FUNCTION:     1915.00428
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2807139E+02     0.1914799E+04
     0.0000000E+00     0.2238172E+01     0.1516912E+03
     0.0000000E+00     0.6664002E-07     0.3830577E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   26674.8591
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4272006E+04     0.2633055E+05
     0.0000000E+00     0.5344547E+03     0.3317098E+04
     0.0000000E+00     0.2040316E+02     0.6317212E+02
```

– Second iteration:

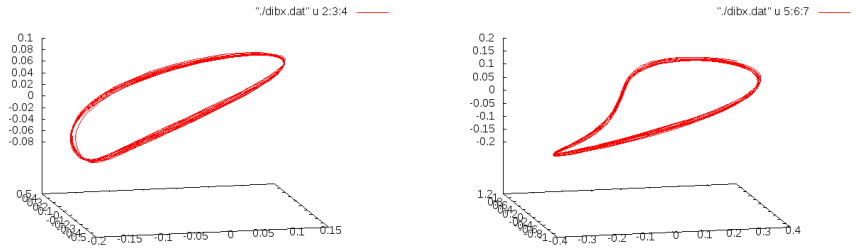

FIG. 17. Parallel shooting in L_2 for Io-Jupiter.

```
NORM OF THE FUNCTION:     316.268971
```

```
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4648145E+01     0.3162348E+03
     0.0000000E+00     0.1337852E+01     0.9051012E+02
     0.0000000E+00     0.1228781E-06     0.3670080E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   624.49512
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.8721231E+02     0.6183754E+03
     0.0000000E+00     0.9563989E+01     0.1326547E+03
     0.0000000E+00     0.1238919E+01     0.3646977E+01
```

– Last iteration:

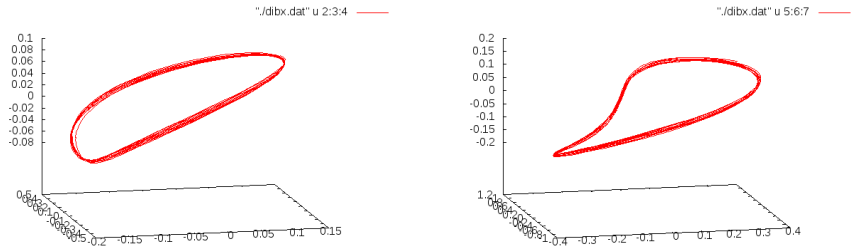

FIG. 18. Parallel shooting in L_2 for Io-Jupiter.

```
NORM OF THE FUNCTION:     6.2390042E-06
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4383272E-05     0.4439831E-05
     0.0000000E+00     0.7257340E-06     0.1529473E-05
     0.0000000E+00     0.0000000E+00     0.2226122E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   8.05399824E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1387084E-04     0.7933655E-04
     0.0000000E+00     0.2037532E-05     0.1139317E-04
     0.0000000E+00     0.7280142E-07     0.3100533E-06
```

- Now, we take Jupiter with his satellite Europa (`ICO=502`), $\mu = 0.2528017529000000 \cdot 10^{-04}$ and for the Lagrange point L_1, $\gamma = 0.2021061523651372 \cdot 10^{-01}$ (amplitude 0.01):

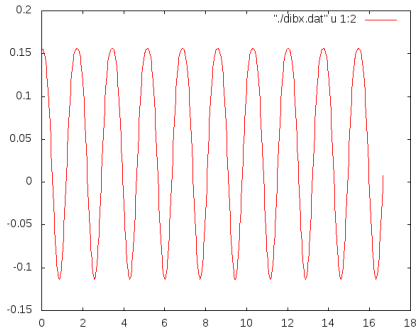

FIG. 19. Time used at parallel shooting in L_1 for Europa-Jupiter.

– First iteration:

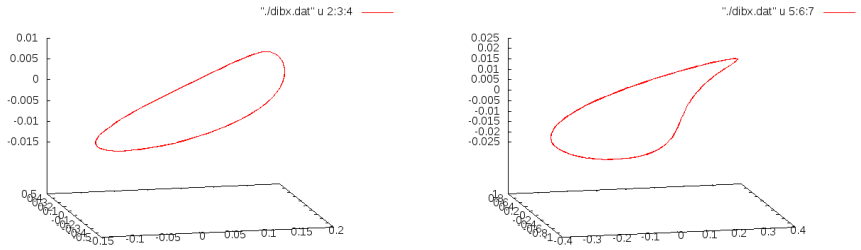

FIG. 20. Parallel shooting in L_1 for Europa-Jupiter.

```
NORM OF THE FUNCTION:     1100.37908
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3153068E+02     0.1099927E+04
     0.0000000E+00     0.3222446E+01     0.1117824E+03
     0.0000000E+00     0.5960464E-07     0.7070158E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   9591.78432
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2789491E+04     0.9177204E+04
     0.0000000E+00     0.3848374E+03     0.1205722E+04
     0.0000000E+00     0.2373108E+01     0.1946122E+02
```

– Second iteration:

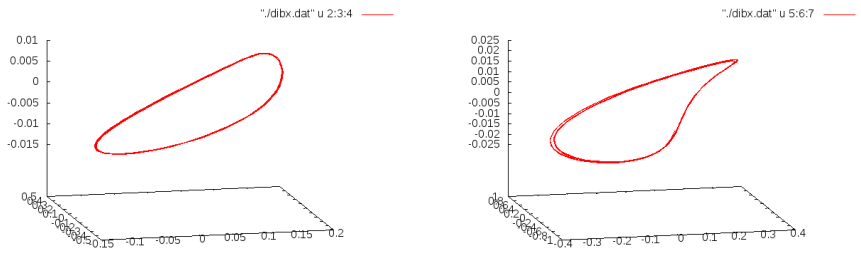

FIG. 21. Parallel shooting in L_1 for Europa-Jupiter.

```
NORM OF THE FUNCTION:     58.5017755
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1668791E+01     0.5847797E+02
     0.0000000E+00     0.4811927E+00     0.1669594E+02
     0.0000000E+00     0.5960464E-07     0.3944990E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   84.0559423
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2034713E+02     0.8155609E+02
     0.0000000E+00     0.3077272E+01     0.1709398E+02
     0.0000000E+00     0.6335030E-01     0.2622111E+00
```

– Last iteration:

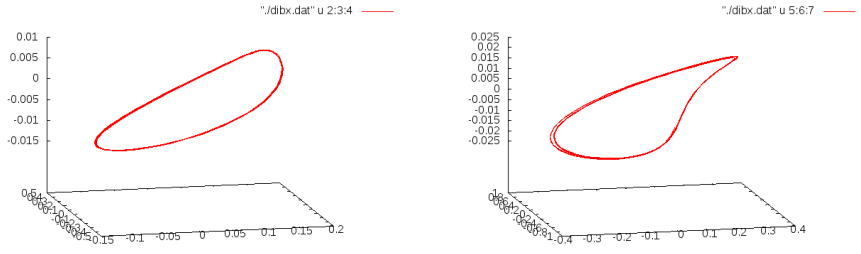

FIG. 22. Parallel shooting in L_1 for Europa-Jupiter.

```
NORM OF THE FUNCTION:      4.65557714E-06
NORM in TIME, POS and VEL, MAX and MIN VALUES:
      0.0000000E+00      0.4170196E-05      0.2069750E-05
      0.0000000E+00      0.6056548E-06      0.5688227E-06
      0.0000000E+00      0.0000000E+00      0.9093073E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   5.10955991E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
      0.0000000E+00      0.1503886E-04      0.4883229E-04
      0.0000000E+00      0.2213957E-05      0.7663341E-05
      0.0000000E+00      0.7099970E-07      0.3094536E-06
```

If we change the Lagrange point to L_2, the value of γ is $\gamma = 0.2048666615815237 \cdot 10^{-01}$, and we change the amplitude to 0.05:

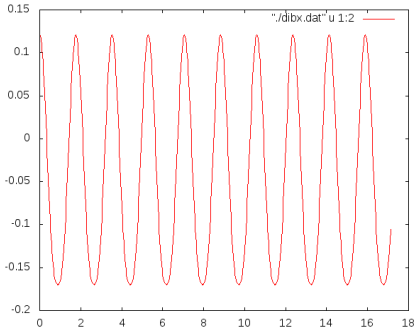

FIG. 23. Time used at parallel shooting in L_2 for Europa-Jupiter.

– First iteration:

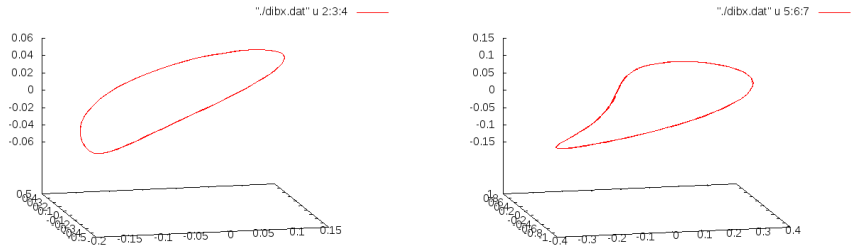

FIG. 24. Parallel shooting in L_2 for Europa-Jupiter.

```
NORM OF THE FUNCTION:     1213.97315
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3578549E+02     0.1213446E+04
     0.0000000E+00     0.3806867E+01     0.1286054E+03
     0.0000000E+00     0.0000000E+00     0.1017599E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   7974.18252
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2277481E+04     0.7642033E+04
     0.0000000E+00     0.3203926E+03     0.1080470E+04
     0.0000000E+00     0.1118441E+02     0.1614795E+02
```

– Second iteration:

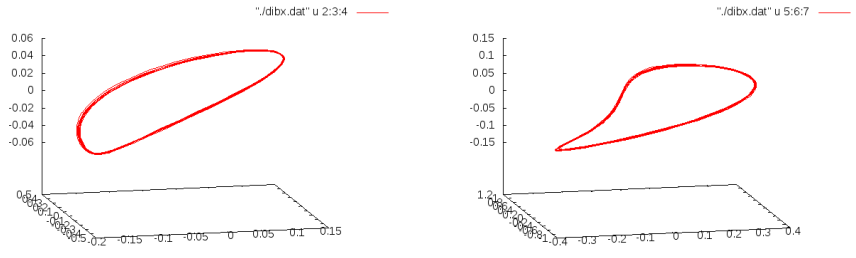

FIG. 25. Parallel shooting in L_2 for Europa-Jupiter.

```
NORM OF THE FUNCTION:     33.2875169
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.9766608E+00     0.3327319E+02
     0.0000000E+00     0.3289830E+00     0.1129438E+02
     0.0000000E+00     0.0000000E+00     0.1641910E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   113.750367
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3647014E+02     0.1077454E+03
     0.0000000E+00     0.5779851E+01     0.2036167E+02
     0.0000000E+00     0.1918062E+00     0.5378395E+00
```

– Last iteration:

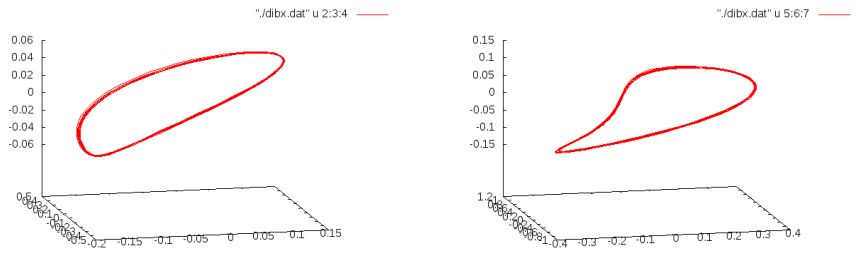

FIG. 26. Parallel shooting in L_2 for Europa-Jupiter.

```
NORM OF THE FUNCTION:     4.78185852E-06
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4337032E-05     0.2014033E-05
     0.0000000E+00     0.7426700E-06     0.4788686E-06
     0.0000000E+00     0.0000000E+00     0.6193716E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   6.73618197E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2171913E-04     0.6376436E-04
     0.0000000E+00     0.2560202E-05     0.7415889E-05
     0.0000000E+00     0.3146396E-06     0.3381366E-06
```

- Taking Saturn with his satellite Titan (`ICO=606`), in L_1, $\mu = 0.2366393209000000 \cdot 10^{-03}$, $\gamma = 0.4226718765727411 \cdot 10^{-01}$ and with amplitude 0.02:

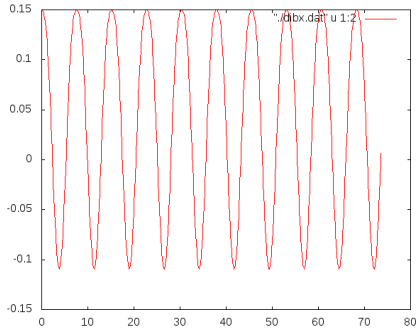

FIG. 27. Time used at parallel shooting in L_1 for Titan-Saturn.

– First iteration:

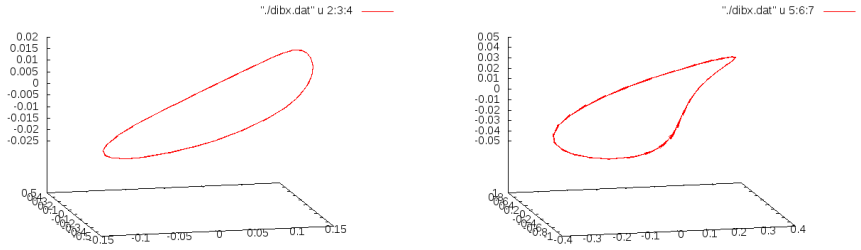

FIG. 28. Parallel shooting in L_1 for Titan-Saturn.

```
NORM OF THE FUNCTION:      2598.33894
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3272154E+03     0.2577653E+04
     0.0000000E+00     0.3766018E+02     0.2967870E+03
     0.0000000E+00     0.1332800E-06     0.4884244E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   24966.1821
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1903281E+05     0.1615743E+05
     0.0000000E+00     0.2261105E+04     0.1624602E+04
     0.0000000E+00     0.4715219E+02     0.1495331E+03
```

– Last iteration:

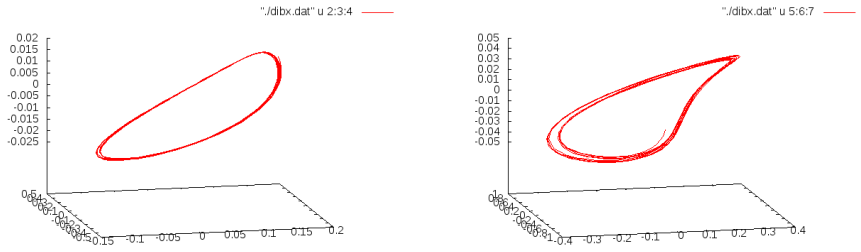

FIG. 29. Parallel shooting in L_1 for Titan-Saturn.

```
NORM OF THE FUNCTION:      6.43897068E-06
```

```
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.6316686E-05     0.1248926E-05
     0.0000000E+00     0.1175587E-05     0.3527809E-06
     0.0000000E+00     0.0000000E+00     0.3790320E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   3.2381252E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2622020E-04     0.1900123E-04
     0.0000000E+00     0.3653660E-05     0.2360504E-05
     0.0000000E+00     0.1561132E-06     0.2194452E-06
```

In L_2, with $\gamma = 0.4349306097794173 \cdot 10^{-01}$ and amplitude 0.06:

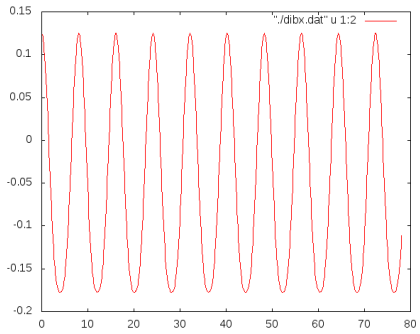

FIG. 30. Time used at parallel shooting in L_2 for Titan-Saturn.

– First iteration:

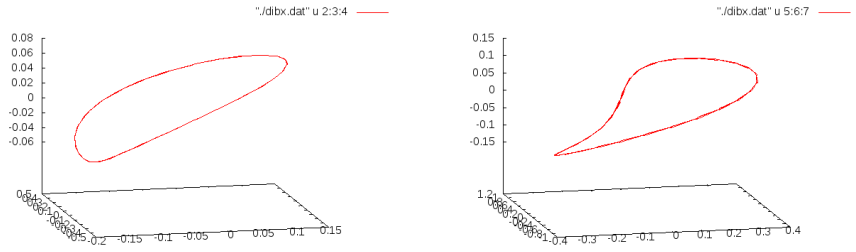

FIG. 31. Parallel shooting in L_2 for Titan-Saturn.

```
NORM OF THE FUNCTION:     2738.41267
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3653451E+03     0.2713932E+04
```

```
     0.0000000E+00     0.3937384E+02     0.2919995E+03
     0.0000000E+00     0.1192093E-06     0.1584489E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   26737.0832
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2037314E+05     0.1731493E+05
     0.0000000E+00     0.2644749E+04     0.1800929E+04
     0.0000000E+00     0.7012327E+02     0.1938728E+03
```

– Last iteration:

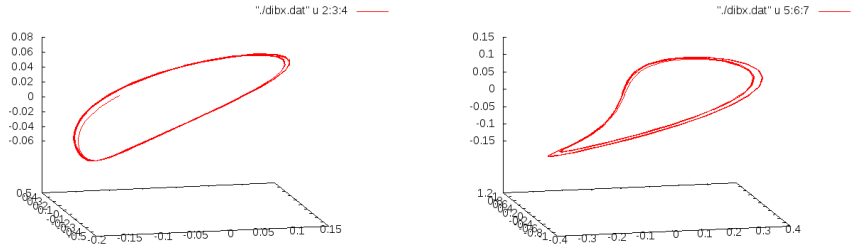

FIG. 32. Parallel shooting in L_2 for Titan-Saturn.

```
NORM OF THE FUNCTION:     6.66645574E-06
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.5928790E-05     0.3048128E-05
     0.0000000E+00     0.8344650E-06     0.1029019E-05
     0.0000000E+00     0.5960464E-07     0.4533196E-08
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   3.89973266E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.3245300E-04     0.2162393E-04
     0.0000000E+00     0.4321357E-05     0.2877422E-05
     0.0000000E+00     0.2915433E-06     0.2421597E-06
```

- Now, we build the orbit for Uranus and the satellite Ariel (`ICO=701`). For the Lagrange point L_1, with $\mu = 0.1492727989000000 \cdot 10^{-04}$, $\gamma = 0.1697443733680036 \cdot 10^{-01}$ and amplitude 0.1, it obtains:

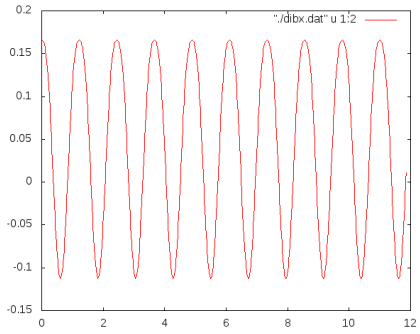

FIG. 33. Time used at parallel shooting in L_1 for Ariel-Uranus.

– First iteration:

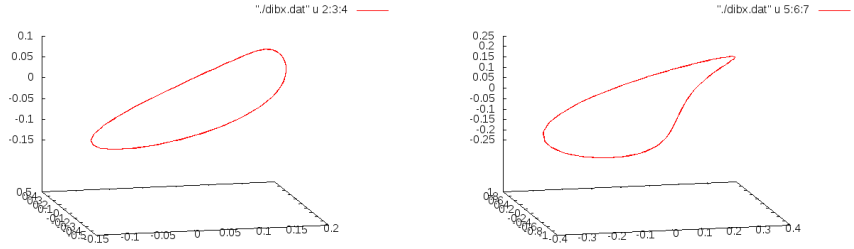

FIG. 34. Parallel shooting in L_1 for Ariel-Uranus.

```
NORM OF THE FUNCTION:      94.4836796
NORM in TIME, POS and VEL, MAX and MIN VALUES:
      0.0000000E+00      0.1918704E+01      0.9446420E+02
      0.0000000E+00      0.1778680E+00      0.8747047E+01
      0.0000000E+00      0.2384186E-06      0.2593882E-06
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   477.824533
NORM in TIME, POS and VEL, MAX and MIN VALUES:
      0.0000000E+00      0.1033608E+03      0.4665113E+03
      0.0000000E+00      0.1551180E+02      0.6794641E+02
      0.0000000E+00      0.2043563E+00      0.4148731E+01
```

– Second iteration:

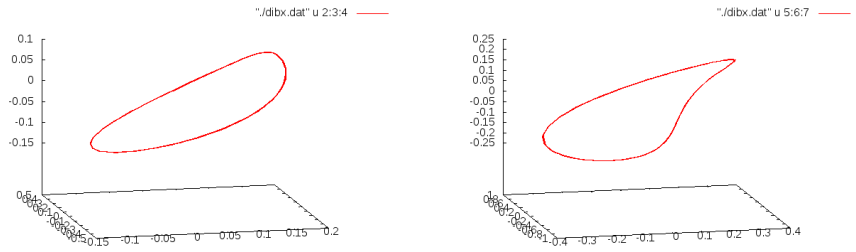

FIG. 35. Parallel shooting in L_1 for Ariel-Uranus.

```
NORM OF THE FUNCTION:      0.433965424
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.8820951E-02     0.4338758E+00
     0.0000000E+00     0.2518004E-02     0.1249481E+00
     0.0000000E+00     0.4298152E-06     0.1557911E-06
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   2.1018556
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4818623E+00     0.2045875E+01
     0.0000000E+00     0.8245416E-01     0.3477684E+00
     0.0000000E+00     0.2837661E-02     0.7021259E-02
```

– Last iteration:

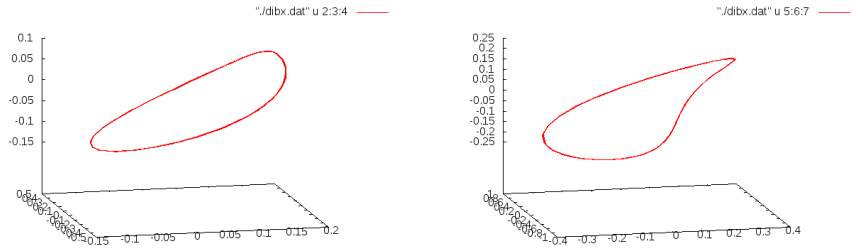

FIG. 36. Parallel shooting in L_1 for Ariel-Uranus.

```
NORM OF THE FUNCTION:      2.20734773E-05
```

```
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1561960E-04     0.1559700E-04
     0.0000000E+00     0.2570612E-05     0.4708597E-05
     0.0000000E+00     0.0000000E+00     0.5095807E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   0.000309964666
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.6208647E-04     0.3036830E-03
     0.0000000E+00     0.8198286E-05     0.4554202E-04
     0.0000000E+00     0.1518095E-06     0.1249099E-05
```

In L_2, with $\gamma = 0.1716873324576420 \cdot 10^{-01}$ and amplitude 0.03 we obtain:

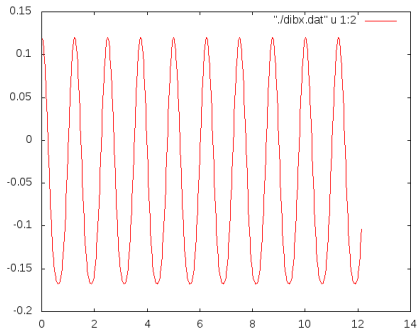

FIG. 37. Time used at parallel shooting in L_2 for Ariel-Uranus.

– First iteration:

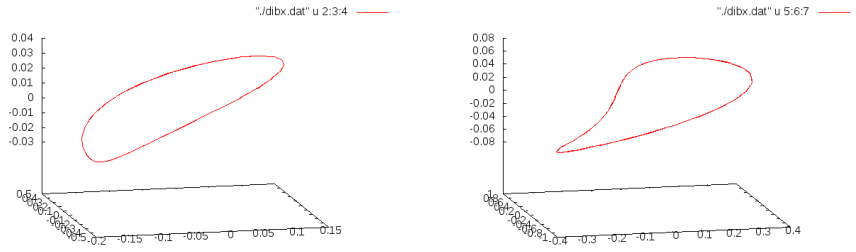

FIG. 38. Parallel shooting in L_2 for Ariel-Uranus.

```
NORM OF THE FUNCTION:     98.6572669
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.2054024E+01     0.9863588E+02
```

```
     0.0000000E+00     0.2118500E+00     0.1014417E+02
     0.0000000E+00     0.0000000E+00     0.1984822E-06
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   1804.01668
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.4067509E+03     0.1757564E+04
     0.0000000E+00     0.5423723E+02     0.2092235E+03
     0.0000000E+00     0.1878637E+01     0.6616225E+01
```

– Second iteration:

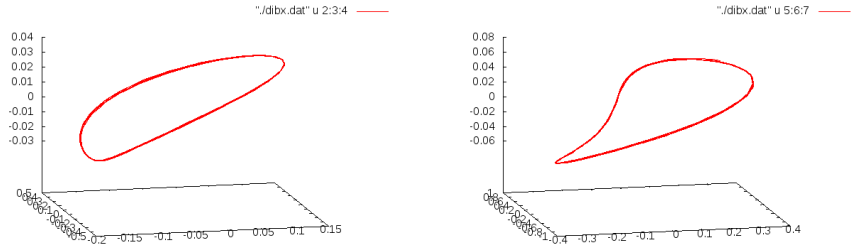

FIG. 39. Parallel shooting in L_2 for Ariel-Uranus.

```
NORM OF THE FUNCTION:     6.72014517
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1398830E+00     0.6718689E+01
     0.0000000E+00     0.4347454E-01     0.2119937E+01
     0.0000000E+00     0.1192093E-06     0.3900060E-06
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   10.3430768
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1985713E+01     0.1015067E+02
     0.0000000E+00     0.3754243E+00     0.2472699E+01
     0.0000000E+00     0.1067216E-01     0.4354275E-01
```

– Last iteration:

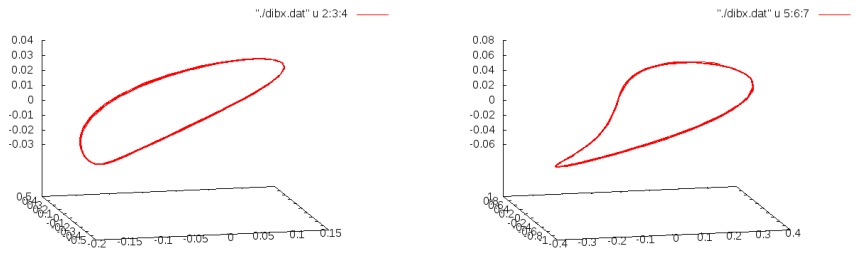

FIG. 40. Parallel shooting in L_2 for Ariel-Uranus.

```
NORM OF THE FUNCTION:     1.98769928E-05
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.1644311E-04     0.1116777E-04
     0.0000000E+00     0.2514723E-05     0.2114996E-05
     0.0000000E+00     0.0000000E+00     0.4886544E-07
PARAMETER IPES= 0
 CORRECTION NORM COMPUTED   WITHOUT  WEIGHTS
NORM OF THE CORRECTION:   0.000283452823
NORM in TIME, POS and VEL, MAX and MIN VALUES:
     0.0000000E+00     0.6175974E-04     0.2766428E-03
     0.0000000E+00     0.7557017E-05     0.3541287E-04
     0.0000000E+00     0.5856849E-06     0.1957551E-05
```

Chapter 5
Conclusions

The present project has kept to build a software package which will be useful in the future for other purposes. Until now, it didn't a software that could read ephemerides files and allowed manage the list of bodies in function of the task that the user is going to perform.

The first that we noted, is that ephemerides files have a very particular format to be read. These files are in a "especial binary format", for which it is necessary use the library `spicelib` offered by JPL. This library is a bit difficult to use, because, although is well commented, has a lot of functions to do similar or different tasks.

Talking about the software, the first that we did, once we could read the files ephemerides, was create several routines to manage bodies, as well as to choose what function have the bodies in the system. After of this, we were created a routine to calculate position, velocity, acceleration and overacceleration for a determinated body in a determinated time. This routine is very necessary for the all tasks that we wanted to do. In fact, we use this routine in all the routines that do a "mathematical work". Finally, we have created a integration routine (a method Runge-Kutta), and, with this, we had the routines necessaries to can do different calculations.

With this "sub-package" created, we create other programs formed by vector fields that we thought that could be interesting. Moreover, we create other programs to do changes of coordinates between several reference systems.

With all routines, we could do simulations using differents initial conditions. Finally, we applicate this software package to the calculate of real trajectories from obtened in a simplified model.

In this software, we have tried reduce the time of calculations to the maximum possible. And we think that can serve to different calculate for planets and satellites. Also, easily it can add other subroutines with contain, for example, other vector fields or other changes of coordinates.

References

[1] Arora,N., Russell, R.P. "A fast, accurate, and smooth planetary ephemeris retrieval system", Celest Mech Dyn Astr (2010), 108, 107–124.

[2] Estabrook,F.B. (1971a). "Derivation of Relativistic Lagrangian for n-Body Equations containing Relativity Parameters β and γ" JPL IOM, Section 328, 14 June 1971.

[3] Estabrook,F.B. (1971b). private communication.

[4] Gomez,G., Masdemont,J.J., Mondelo,J.M. "Solar system models with a selected set of frequencies", Astronomy and Astrophysics (2002).

[5] Hou,X.Y., Liu,L. "On quasi-periodic motions around the triangular libration points of the real Earth–Moon system", Celest Mech Dyn Astr (2010), 108, 301–313.

[6] Hou,X.Y., Liu,L. "On quasi-periodic motions around the collinear libration points in the real Earth–Moon system", Celest Mech Dyn Astr (2011), 110, 71–98.

[7] Ma,C., Arias,E.F., Eubanks,T.M., Fey,A.L., Gontier,A.-M., Jacobs,C.S., Sovers,O.J., Archinal,B.A., and Charlot,P. (1998). "The International Celestial Reference Frame as Realized by Very Long Baseline Interferometry", Astron. J., 116, 516-546.

[8] Newhall,X X, Standish,E.M. and Williams,J.G. (1983). "DE102: a numerically integrated ephemeris of the Moon and planets spanning forty-four centuries", Astron. Astrophys. 125, 150-167.

[9] Seidelmann,P.K.."Explanatory supplement to the astronomical almanac", U.S. Naval Observatory, Washington, D.C., University Science Books.

[10] Standish,E.M. "JPL Planetary and Lunar Ephemerides, DE405/LE405", Jet Propulsion Laboratory (1998), IOM 312.F-98-048.

[11] Will,C.M. (1974). "Experimental Gravitation B.Bertotti", ed. (Academic Press, New York).

[12] "Introduction to the SPICE Ephemeris Subsystem SPK - Focused on reading SPK files", `http://naif.jpl.nasa.gov/naif/tutorials.html`.

[13] http://naif.jpl.nasa.gov/naif/toolkit_FORTRAN_PC_Linux_g77_32bit.html

[14] ftp://naif.jpl.nasa.gov/pub/naif/generic_kernels/spk/

[15] http://ssd.jpl.nasa.gov/